UNIVERSIDAD AUTÓNOMA DE MADRID

Escuela Politécnica Superior

Doble Grado en Ingeniería Informática y Matemáticas

TRABAJO FIN DE GRADO

UN ENFOQUE DE ROBÓTICA VIRTUAL PARA PERSONAS QUE TIENEN DISCAPACIDAD MOTORA

Cristina Kasner Tourné

Tutor: Francisco de Borja Rodríguez Ortiz

Enero 2017

UN ENFOQUE DE ROBÓTICA VIRTUAL PARA PERSONAS QUE TIENEN DISCAPACIDAD MOTORA

Autor: Cristina Kasner Tourné
Tutor: Francisco de Borja Rodríguez Ortiz

Escuela Politécnica Superior
Universidad Autónoma de Madrid

Enero 2017

RESUMEN

Resumen La fusión de la Realidad Virtual con la robótica supone una apertura a infinitas posibilidades.

Ambas tecnologías están en continuo desarrollo, de hecho la realidad virtual ha vuelto a gozar de popularidad muy recientemente, a pesar de que su nacimiento data de la década de los 60.

En los últimos años estamos presenciando cómo la Realidad Virtual va haciéndose un hueco en las tecnologías que usamos habitualmente. Quizá el uso más comercial que se le está dando es en lo referente al mundo de los videojuegos. Sin embargo se puede aplicar a muchos otros campos, en concreto al campo de la salud, donde se están obteniendo muy buenos resultados.

Este trabajo pretende explorar el uso de la Realidad Virtual y la robótica como herramienta enfocada a personas con movilidad reducida, ayudarles a ser más independientes dándoles la posibilidad de transportarse de forma virtual por entornos reales.

Para conseguir este objetivo se ha construido un sistema que consta de un **robot** y las **Oculus Rift** (gafas de realidad virtual). El robot se controla de forma inalámbrica desde las Oculus Rift, así si el usuario lleva puestas las gafas, podrá controlar los movimientos del robot (de transporte y de exploración) con movimientos de cabeza y verá a través de las gafas todo lo que vea el robot, actuando éste último como extensión de la vista del usuario.

Una vez que se terminó de construir el robot e implementar el software, se realizaron pruebas con diferentes usuarios para valorar su usabilidad. En base a estas pruebas se rediseñó la forma de enviar comandos y se ajustaron los parámetros pertinentes.

Una vez terminados todos los cambios se realizaron unas últimas pruebas a otros usuarios distintos. Los resultados son muy satisfactorios ya que en pocos minutos la sensación de control de los usuarios aumenta y les cuesta mucho menos alcanzar los objetivos propuestos.
Aún cuando el proyecto está orientado a personas con discapacidad motora, no se ha podido probar con ninguna persona en esta situación. Por ello es de gran utilidad que los parámetros de velocidad, rango de movimientos, sensibilidad, etc. sean parámetros ajustables en el sistema. De esta forma, cuando en un trabajo futuro se pruebe para los

usuarios a los que está enfocado el proyecto, será sencillo calibrarlo y adaptarlo a las necesidades pertinentes.

A pesar de las valoraciones positivas, no es recomendable utilizar las gafas durante mucho tiempo seguido si el usuario no está acostumbrado a ellas ya que en algunos casos puede causar mareos y en los últimos años se cuestiona el impacto que pueden tener las gafas de realidad virtual en la retina de los usuarios [1].

El resultado final del trabajo es un dispositivo móvil que el usuario puede controlar sólo con la cabeza y que le permite visualizar su entorno a través de las Oculus Rift. Al utilizar las Oculus Rift dejamos abierta la posibilidad de ver en un trabajo futuro, no solo el entorno en el que se mueve el robot, sino un entorno virtual creado por el propio usuario.

Palabras clave realidad virtual, Oculus Rift, discapacidad motora, robótica.

Abstract

Abstract The fusion between Virtual Reality and robotics brings a world of infinite possibilities.

Both technologies are under continuous development. In fact, Virtual Reality has regained popularity very recently even though it was born during the nineteen sixties.

In recent years we have witnessed how Virtual Reality finds a spot amongst our most used technologies. The most widespread commercial use comes from home entertainment, games concretely. Nevertheless it can be applied to many other different fields, like health, in which outstanding results are being obtained (in surgery, rehabilitation, etc).

This project attempts to explore the use of Virtual Reality and robotics as a tool focused on individuals who are disadvantaged with reduced mobility, to help them be more independent and bringing them the possibility of virtual immersion and transportation in real environments.

To achieve this goal a system composed of a **robot** and the **Oculus Rift** glasses (Virtual Reality) has been built. The robot is controlled by the Oculus Rift glasses. A user wearing them will be able to control the robot's movements (transportation and exploration) with head movements. The user will see in real time through the glasses what the robot is seeing, making the robot an extension to the user's vision.

Once the robot was completely built and the software fully implemented, different usability tests were performed with several users. Based on these usability tests the way to send commands was redesigned and the related parameters were adjusted.

Those first results led to the idea of a new control interface which, after implemented, resulted in better usability. It is clear how during the first attempts at controlling the robot the users were struggling to control the robots movements with precision, as they were not used to controlling movement just with their head's position. After this round of trials the robot's speed was adjusted to make it slightly slower.

Once all changes were done, the final tests were performed with new test users. These results were very positive as users would quickly get accustomed to the robots sensitivity and movements after a few minutes, enabling them to reach the proposed targets much easier.
Even though the project is focused on people suffering from motor disabilities, it hasn't

been able to test it with anyone with that condition. That's why it is important to have speed, range of movement, sensitivity, etc... as adjustable parameters in the system. This way when, during future work, it is tested with the users this project is targeted for it will be easy to calibrate and customize for that case's needs.

Despite the positive reviews, it is not recommended to spend a lot of time using the system without taking the glasses off, specially if the user is not accustomed to them. In some cases it may cause dizziness and it has been questioned whether Virtual Reality Glasses may be damaging the users' retinas [1].

This project's final result is a movil device that the user can control just by using their heads and which enables them to see the device's environment through Ouclus Rift. By using Oculus Rift we keep the possibility of seeing, not only the environment inside which the robot is moving, but a virtual environment created by the user himself. These questions have been left as study material for future work.

Keywords virtual reality, Oculus Rift, motor disabilities, robotics

AGRADECIMIENTOS

Este trabajo no hubiera sido posible sin todas las personas que me han acompañado y guiado a lo largo de estos meses y también durante toda la carrera.

Quiero agradecer a todos los miembros del Club de Robótica el haberme abierto las puertas del club y haberme enseñado prácticamente todo lo que sé sobre robótica. Quiero agradecer en especial a Carlos, a Javi, a Pablo, a Jaime, a los Guilles y a los Victors por toda la paciencia que tuvieron cuando decidí empezar con todo esto.

También quiero agradecer a todo mi curso del doble grado estos años de carrera, han sido unos compañeros estupendos y han hecho mucho más fácil esta etapa de estudios.

Quiero agradecer a mi tutor, Francisco de Borja, la oportunidad de introducirme en el mundo de la realidad virtual y poder aprender a manejar las Oculus Rift. Quiero agradecerle la paciencia que ha tenido durante este trabajo, con los correos, las fechas... y en general el apoyo que me ha dado.

Por último quiero agradecer a mi familia, que se sienten orgullosos con poquito que hago, que me apoyan en todo y me motivan a buscar aplicaciones más sociales a lo aprendido en la carrera.
A todos ellos y a los que no he nombrado, gracias.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 Introducción

1.1 Motivación del proyecto

Este proyecto busca explorar las posibilidades que nos ofrece combinar la robótica con la realidad virtual. En concreto pretende estudiar la capacidad que la fusión de estas tecnologías tiene para incrementar la autonomía e independencia de personas con discapacidad motora.

Se ha investigado qué proyectos anteriores combinan robótica y realidad virtual y se han encontrado un gran número de ellos, muchos con resultados exitosos, como ya se explicará en el estado del arte. A pesar de que algún proyecto anterior ya estaba enfocado al tema de la discapacidad, la gran mayoría se centran en intentar estimular a estas personas y no en tratar de hacer que su entorno sea más accesible.

Una de las ventajas de hacer un proyecto basado en robótica y realidad virtual es que ambas tecnologías tienen un gran potencial [4]. El campo de la **robótica** lleva muchos años en desarrollo y mejora exponencialmente a lo largo del tiempo. Está inmerso en nuestro día a día y no paramos de sorprendernos con nuevos avances como las impresoras 3D, los drones o los brazos mecánicos. Por otra parte puede parecer que la **realidad virtual** es una tecnología muy reciente pero lo cierto es que lleva mucho tiempo estudiándose. En 1968 Ivan Sutherlan creó el primer casco de realidad virtual pero, como se explicará más adelante, aún no tenían las herramientas necesarias para desarrollar proyectos de realidad virtual de calidad.

Ahora nos encontramos en un momento en el que ya podemos empezar a disfrutar de la realidad virtual. Tenemos las CardBoard, las Oculus Rift, etc, a nuestro alcance. El uso más generalizado que se está dando a estas herramientas es en el mundo de los videojuegos. Viendo el potencial que tiene unir ambas tecnologías, resulta muy interesante explorar nuevas aplicaciones, en esta ocasión orientadas a mejorar la calidad de vida de un amplio sector de la sociedad.

La realidad virtual nos ofrece la posibilidad de **crear un entorno** en el cual, con el software adecuado, el usuario puede **interactuar** con otros usuarios, participar en actividades de **ocio y entretenimiento**, etc, sin necesidad de moverse de casa. Además puede resultar muy interesante utilizar esta tecnología para **reproducir virtualmente entornos reales** a los que el usuario tenga difícil acceso.

El primer paso para conseguir esto es introducir imágenes del entorno en tiempo real dentro del mundo virtual. Para ello debemos disponer de un mecanismo móvil(p.ejemplo: un robot) que sea capaz de capturar fotogramas y mandárselas al usuario. Para que el usuario pueda ver dichas imágenes se utilizarán las Oculus Rift, unas potentes gafas de realidad virtual que son capaces de crear una **sensación de inmersión** total en cualquier entorno. Ya que este trabajo está orientado a personas con movilidad reducida sería ideal que se pudiera controlar el movimiento del robot y de la cámara solo con el movimiento de la cabeza. En un futuro, no sería difícil adaptar este control a través de mandos.

¿Porqué utilizar las Oculus Rift? Al ser las Oculus gafas de realidad virtual,si consiguiéramos una correcta conexión entre las gafas y el robot, se nos abrirían muchísimas posibilidades. Se podría modificar el entorno según las necesidades del usuario, integrar realidad aumentada para controlar una casa domotizada, incluir en el mundo virtual más entornos, a parte del real, donde el usuario pueda interactuar, realizar actividades de ocio, etc. De esta forma daríamos al usuario más independencia y le abriríamos un mundo al que hasta ahora, tiene difícil acceso.

1.2 Objetivos

El objetivo de este proyecto consiste en combinar herramientas de robótica y de realidad virtual para diseñar un dispositivo que actúe como extensión de la vista del usuario, a través de una webcam implantada en el robot. El usuario deberá ser capaz de controlar la los movimientos de cámara con tan solo el movimiento de la cabeza, ya que el proyecto está orientado a personas con movilidad reducida.
La cámara, gracias a estar implantada en el robot (dispositivo móvil), tendrá la capacidad de realizar movimientos de rotación respecto al eje vertical, rotación respecto al eje horizontal y movimientos de desplazamiento.
Las imágenes del entorno del robot se reproducirán a través de unas gafas de realidad virtual: las Oculus Rift. Esto nos permitiría, en un proyecto futuro, añadir realidad virtual o aumentada a las imágenes que captura la cámara.

Para ello definimos pequeños objetivos que debemos ir implementando para conseguir la meta final: streaming, movimiento de la cámara/robot (de rotación y de transporte) y visualización del entorno.

1.2.1 Streaming

Como se ha explicado anteriormente, debemos ser capaces de ver en las Oculus Rift a tiempo real todo lo que capta la cámara.

Dado que la comunicación con las Oculus Rift debe ser a través del PC al que estén conectadas, el streaming deberá **transmitirse desde la cámara hasta el PC**.

Por lo tanto uno de los primeros objetivos es implementar un método de comunicación a través del cual se puedan conectar la cámara y el PC. Además se deberá implementar un software que capture la salida de la webcam y transmita el vídeo a tiempo real, es decir, que el usuario pueda saber con la mayor exactidud posible, el entorno que rodea al robot en todo momento. Por último se necesita un software que recoja el vídeo en el PC.

1.2.2 Movimiento de la cámara

Otro de los objetivos básicos es que el usuario sea capaz de controlar la dirección y posición de la cámara con el movimiento de la cabeza. Para ello dotaremos a la cámara de un soporte móvil, un robot, que se explicará en el capítulo de Análisis.

Para controlar la cámara definimos dos fases del movimiento:

- **Fase de exploración:** Durante la exploración la cámara no se mueve de sitio, tan solo explora el entorno. Los movimientos que se han decidido implementar para esta fase serán:
  - Movimiento vertical : El usuario podrá mirar hacia arriba y hacia abajo.
  - Movimiento horizontal : La cámara deberá ser capaz de rotar sobre si misma permitiendo al usuario mirar hacia ambos lados.
- **Fase de transporte:** El usuario, enviando los comandos pertinentes al robot, podrá llevar la cámara a cualquier punto de la habitación.

Para este objetivo, el diseño tendrá un papel esencial ya que el control del robot debe ser sencillo e intuitivo. Se intentará crear un sistema de control con la cabeza que no sea complicado para el usuario.

1.2.3 Visualización del entorno

El último de los objetivos que es necesario desarrollar para el proyecto es conseguir visualizar en las Oculus Rift, el entorno en el que se encuentra la cámara. Para ello necesitaremos un software que, en tiempo real, recoja el vídeo del PC y lo reproduzca en las gafas.

A continuación se añade un esquema a modo de resumen de los objetivos ya explicados.

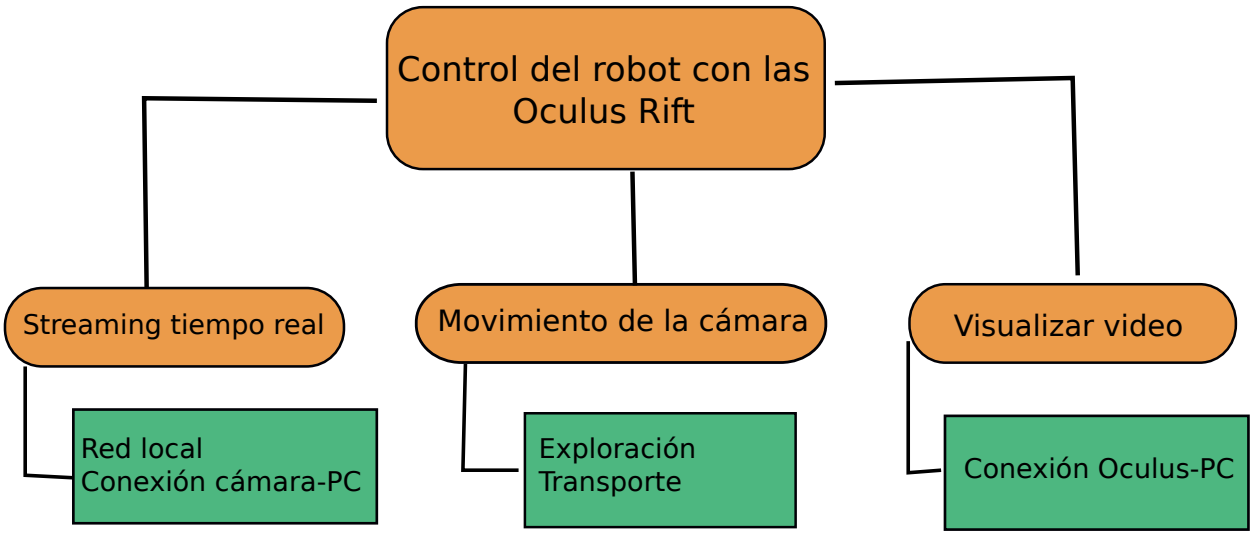


Figura 1.1: Resumen de los objetivos del proyecto

1.3 Desarrollo y plan de trabajo

El desarrollo de este proyecto de fin de grado se puede dividir en tres fases:

- **Investigación y aprendizaje:** Durante esta primera fase me documenté acerca de la realidad virtual, de sus aplicaciones y de las herramientas de las que disponía para realizar el trabajo. También busqué proyectos parecidos que unieran la robótica y la realidad virtual para saber qué placas funcionaban mejor con gafas de realidad virtual.
- **Diseño y desarrollo:** Una vez adquirida la base de conocimiento necesaria para empezar el proyecto pude empezar la fase dos, que incluye:
  - Diseño y construcción del robot
  - Diseño e implementación del software
- **Pruebas:** Durante la fase de pruebas un grupo de usuarios utilizó y valoró el proyecto. Además se hicieron cambios de diseño en base a la valoración de los usuarios .

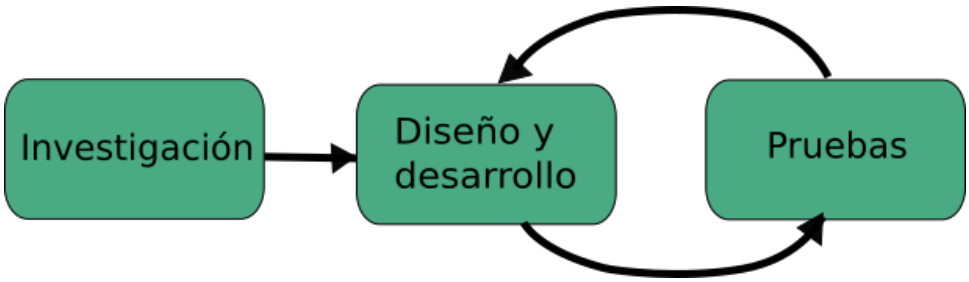


Figura 1.2: En este grafo queda resumido el plan de trabajo. Se puede ver cómo la fase de pruebas vuelve a la fase de diseño y desarrollo, ya que el proyecto va mejorando en base a lo que demandan los usuarios.

ESTADO DEL ARTE

2

En este capítulo se contará brevemente la historia de la realidad virtual. También se presentarán diferentes proyectos que han utilizado la combinación de realidad virtual y robótica y cuales fueron sus resultados.

La cara más visible de esta tecnología es el uso que se le da en los videojuegos [5], pero no es su única aplicación. Los proyectos que aquí se exponen están orientados a la salud o al control de dispositivos, ya que son más afines con este trabajo fin de grado. Los proyectos relativos al campo de la salud, se aplican mayoritariamente a rehabilitaciones. Esto es muy interesante para este trabajo ya que muestra cómo usuarios con alguna dificultad motora responden bien a entornos virtuales [6]. También se hablará de cómo se están integrando actualmente las dos tecnologías que utilizaremos en este trabajo: la robótica y la realidad virtual.

La explicación de estos proyectos en este capítulo pretende dar una fotografía general del estado y las aplicaciones actuales de ambas tecnologías, dando especial importancia a aquellos proyectos con un enfoque similar al de nuestro trabajo.

2.1 Evolución histórica de la realidad virtual

La realidad virtual comenzó siendo ciencia ficción y poco a poco está ganando terreno en la vida cotidiana, ya sea en el mundo de los videojuegos, la medicina o la aviación, entre otros campos.

A pesar de ser una tecnología de la que se ha empezado a hablar hace relativamente poco, los primeros proyectos que utilizaban realidad virtual datan de los años 60.
Uno de los acontecimientos más significativos del inicio de la realidad virtual es la creación de **"The Sword of Damocles"**, un casco de realidad virtual (HMD- head-mounted display) creado por Ivan Sutherland en 1968 [2].
The Sword of Damocles permitía ver las imágenes tridimensionales generadas por el ordenador superpuestas ópticamente, a través de un prisma, al entorno real que rodeaba al usuario. [1]
El sistema estaba montado sobre un brazo mecánico suspendido del techo que también servía para detectar, mediante el uso de sensores, la posición y la orientación de la cabeza del usuario. De este modo, cuando el usuario se movía, los objetos creados por el

[1] En este vídeo se puede ver un ejemplo de The Sword of Damocles: https://www.youtube.com/watch?v=ISJWZpFIAlQ

ordenador daban la impresión de encontrarse en una posición estable dentro del entorno real.

Es lógico preguntarse cómo ha tardado tanto en llegar al mercado la realidad virtual si ya en 1968 consiguieron sacar adelante este proyecto.
La respuesta es que **la tecnología de los años 60 aún no estaba preparada** para desarrollar herramientas de realidad virtual.
Para conseguir capturar la posición y el movimiento de la cabeza del usuario, se requería un brazo de **grandes dimensiones**, como se puede ver en la figura 2.1.
Además la realidad virtual necesita una **capacidad de cómputo muy potente**. Problemas tales como "hidden line" solo podía ser resueltos con un ordenador que tenía la NASA en Houston [7], como especifica Sutherland en su trabajo *A head-mounted three dimensional display* [2].

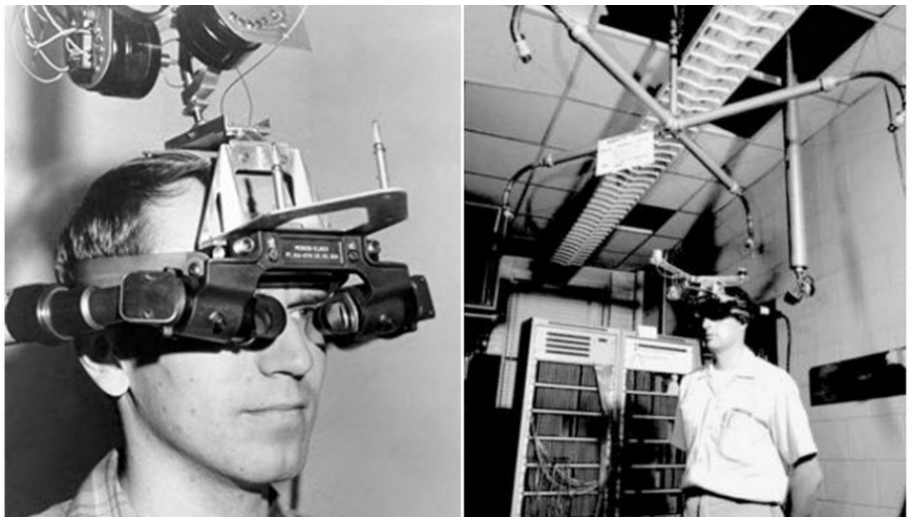


Figura 2.1: Sword of Damocles. Imagen obtenida de *A head-mounted three dimensional display* [2]

Al mismo tiempo Thomas A. Furness empezó a introducir herramientas de **realidad virtual en entrenamientos de vuelo** [8], con el objetivo de que estos fueran mucho más seguros para los pilotos. A pesar de que fue una gran aportación en este campo, la escasez de recursos y capacidad de cómputo también retrasaron mucho el desarrollo de estas herramientas. En la actualidad se considera a Thomas A. Furness [2] uno de los padres de la realidad virtual y la realidad aumentada y tiene un papel muy importante en este campo.

Hoy en día tenemos la capacidad necesaria para desarrollar estas herramientas por lo que se siguen utilizando y mejorando ya que, no solo es más segura, sino que además es más barata.
Durante las últimas décadas del s.XX, se hicieron algunos proyectos de realidad virtual, pero no tuvieron mucho éxito debido a su alto coste.
Con el desarrollo exponencial que ha vivido la tecnología las últimas décadas, la realidad virtual ha ido haciéndose un hueco en la industria tecnológica. El empujón definitivo vino con la creación de las **Oculus Rift**, en el año 2010.

Palmer Luckey es el fundador de Oculus VR (una compañía de realidad virtual) y el diseñador de la primera versión de las Oculus Rift [3].
Oculus Rift es un casco de realidad virtual (también nos podemos referir a ellas como

[2]Charla de Thomas A. Furness https://www.youtube.com/watch?v=zr89F88AuUIThomas A. Furness
[3]Página oficial de las Oculus Rift: https://www.oculus.com/

gafas de realidad virtual) mucho más manejable que "The Sword of Damocles", con mucha más calidad y con un coste asequible. Uno de los grandes logros de Luckey con las Oculus Rift es que consiguió un ángulo de visión de 90º, algo nunca visto hasta el momento.
Tras el lanzamiento de las Oculus Rift, el mercado comenzó a interesarse por proyectos relacionados con la realidad virtual y a financiarlos. Gracias a esto la tecnología ha podido desarrollarse y hoy podemos disfrutar de una realidad virtual con mucha más calidad y **sensación casi total de inmersión** en entornos virtuales.

2.2 Realidad virtual y salud

La realidad virtual ofrece grandes posibilidades en el campo de la medicina. Permite estudiar cómo responde el paciente a distintos estímulos o cómo interactúa con diferentes entornos de forma bastante realista.

Podemos encontrar un ejemplo de esto en un proyecto que se realizó para evaluar el deterioro cognitivo en personas que sufren esclerosis múltiple. Para que dicha evaluación sea precisa se requiere de una gran cantidad de datos sobre la velocidad de procesamiento de la información, atención, etc. Esto se conseguía haciendo múltiples test que no siempre eran fieles a la realidad.
Ahora con la realidad virtual se consiguen crear entornos en los que el usuario tiene tal sensación de inmersión, que su **comportamiento es muy similar al que tendría en el entorno real**. Esto permite hacer un estudio mucho más amplio y realista de los estímulos del paciente y obtener resultados más completos.[9]

Otra aplicación interesante es el uso de realidad virtual en programas de rehabilitación [10] [11]. Se realizó un estudio a personas con hemiparesia, es decir, pacientes que han perdido parcial o totalmente la capacidad motora de las extremidades de uno de los lados de su cuerpo. El proyecto consistía en un software que mostraba por pantalla el movimiento de los pies del usuario y le mandaba ejercicios para mejorar su locomoción. Los resultados de este estudio mostraron que la herramienta de realidad virtual era **efectiva para la rehabilitación de las capacidades motoras** de los pacientes y además es especialmente útil para aquellos que tienen dificultades para trasladarse hasta el hospital.[12]

Los dos proyectos que se han presentado en este capítulo son ejemplos de cómo la realidad virtual ha contribuido a mejorar técnicas que ya se usaban anteriormente.
El siguiente proyecto también está enfocado a mejorar el tratamiento de pacientes pero, en esta ocasión, utiliza una característica propia de la realidad virtual, que es la inmersión en un entorno distinto al que se encuentra el paciente.
Esta capacidad que tiene la realidad virtual es muy útil para personas que se encuentran en un proceso de rehabilitación doloroso.

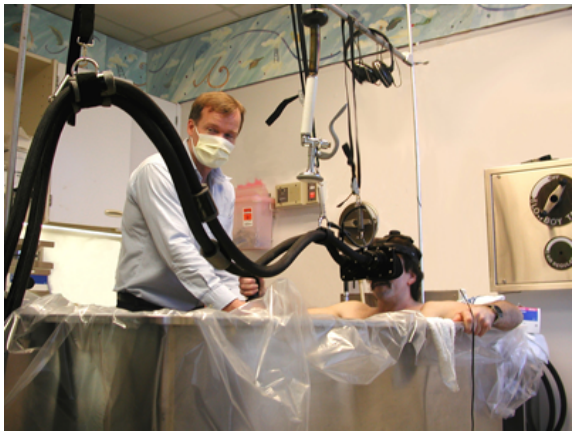


Figura 2.2: Rehabilitación usando herramientas de realidad virtual. Imagen obtenida de *Water-Friendly Virtual Reality Pain Control During Wound Care* [3]

El experimento que se presenta a continuación se realizó con un paciente de 40 años con un 19 % de su cuerpo quemado. El trabajo consistía en utilizar un entorno virtual como paliativo durante los ejercicios de rehabilitación. Se le proporcionó al paciente un casco de realidad virtual y se creó un entorno en el que el paciente pudiera "resguardarse" del dolor ocasionado por la rehabilitación.

Los resultados fueron muy positivos. El usuario puntuó la sensación de dolor en un 7 sobre 10 cuando no usaba realidad virtual y en un 2 sobre 10 utilizando realidad virtual.[3]
Estos experimentos suponen una prueba de la increíble **capacidad de inmersión** que nos proporciona la realidad virtual, aspecto muy importante para nuestro trabajo.

2.3 Robótica y realidad virtual.

A continuación se presentan algunos proyectos de robótica que utilizan técnicas de realidad virtual. Es interesante ver, en los resultados de estos trabajos, cómo ambas tecnologías se complementan de forma muy beneficiosa. [13]

Puesto que al final de este trabajo lo que se quiere conseguir es controlar un robot a través de una interfaz que utiliza realidad virtual, es interesante estudiar cómo ha resultado esto en proyectos parecidos.

En 1997, el IMG (Intelligent Mechanisms Group) de la NASA y el instituto de robótica de la Carnegie Mellon University desarrollaron las **interfaces de control** necesarias para teledirigir a Nomad, un robot diseñado para la exploración de planetas. [14]
Hasta el momento este control era altamente complicado, ya que requería del uso de muchos comandos poco intuitivos. Para mejorar y facilitar esto, crearon dos interfaces: *Virtual Dahsboard Interface* y *Telepresence Interface*.

Virtual Dashboard interface consiste en un panel de control donde el operador podía ver de forma simple el estado actual de Nomad y controlarlo de forma intuitiva.
Telepresence Interface es una interfaz que hace uso de la cámara panorámica de Nomad. A través de dicha interfaz el usuario podía ver las imágenes capturadas por Nomad e incluso podía girar el punto de vista o ampliar la imagen para conocer mejor el entorno.

Estas imágenes se reproducían en una pantalla ancha, parecida a un parabrisas de un coche, frente al operador, para dar más sensación de telepresencia.
Uno de los inconvenientes de este trabajo fue el delay a la hora de recibir las imágenes

(5 segundos), que hacía más difícil el control. A pesar de esto **los resultados fueron positivos**, concluyendo que los operadores manejaban mejor el robot haciendo uso de la Telepresence Interface que sin ella.

Otra de las aplicaciones de la combinación de robótica con realidad virtual es la mejora en **intervenciones quirúrgicas** [15].
Un ejemplo de esto es **Da Vinci Surgical System**, un sistema en el cual el cirujano controla a través de unos mandos, las herramientas necesarias para operar al paciente, tal y como se puede ver en la figula 2.3. Además tiene una pantalla 3D en la que se ve a tiempo real la zona que está siendo intervenida. Este sistema hace posible que el cirujano realice **movimientos mucho más precisos**, además de facilitarle una postura más cómoda durante la operación. [16]

(a) (b)

Figura 2.3: Da Vinci Surgical System

Un proyecto muy reciente y con gran parecido a este trabajo fin de grado es DORA [17], plataforma robótica de inmersión diseñada para la navegación y exploración de lugares remotos. Es un proyecto financiado por varias compañías como Oculus VR o Intel. Como se puede ver es un proyecto con mejor hardware del que se dispone para este trabajo fin de grado. Esto le permite tener mayor grado de libertad y mejor precisión pero el funcionamiento es muy similiar al de este proyecto.

Existen muchos más proyectos de robótica y realidad virtual, como VirBot [18], este proyecto permite a los usuarios controlar un robot, pero a diferencia de nuestro trabajo, este robot se encuentra en un mundo virtual.
También se han estudiado diferentes métodos de controlar un robot en remoto utilizando realidad virtual, un ejemplo de esto es la publicación *Cooperative Robot Teleoperation through Virtual Reality Interfaces* [19], que estudia la mejor forma de definir una interfaz virtual, que a su vez es un escritorio, para controlar un robot.

Lo que nos muestran los resultados de estos proyectos es que los usuarios son capaces de manejar dispositivos que ven a través de realidad virtual de forma exitosa. Esto también es imprescindible en nuestro trabajo ya que el usuario debe controlar el robot tan solo con las gafas de realidad virtual.

ANÁLISIS

3

Una vez conocido el estado actual de la robótica y la realidad virtual, el siguiente paso es analizar a fondo el proyecto que se quiere desarrollar. Para ello se deben definir los requisitos que debe cumplir el trabajo, basando estos requisitos en los objetivos propuestos en la sección de *Objetivos* 1.2. De esta forma el trabajo queda dividido en pequeñas tareas y facilita su implementación.

En la sección de *Objetivos* 1.2 se explica que la meta final del proyecto es conseguir que el usuario pueda controlar una cámara con el movimiento de cabeza y ver todo lo que la cámara vea. La forma más eficiente de hacer que la cámara sea móvil es construir un robot con ruedas que lleve la cámara integrada.

De esta forma cuando el usuario mueva la cabeza, se le enviarán las instrucciones necesarias al robot para que se mueva. A su vez el robot le estará mandando vídeo a tiempo real al usuario, que lo verá a través de las Oculus Rift.

El proyecto por lo tanto se divide en 3 problemas:

1. **Construcción del robot** → para ello necesitaremos una placa base, una cámara y la estructura que le permita desplazarse.
2. **Streaming**→ Debemos conseguir transmitir a tiempo real todo lo que ve el robot a la Oculus
3. **Control del robot**→ Se quiere establecer una equivalencia entre el movimiento de la cabeza del usuario y los movimientos del robot, tanto movimientos de transporte como rotación de la cámara.

Vamos a desarrollar cada uno de ellos, identificando qué funcionalidades básicas debe cumplir.

3.1 Requisitos Funcionales

3.1.1 Construcción del Robot

El robot debe ser un dispositivo fácil de controlar, que a su vez actúe como una extensión de los ojos del usuario, dándole a éste la sensación de inmersión un espacio real. Por tanto los principales requisitos que debe cumplir el robot son los siguientes:

- **RF-CR1: Grabar vídeo.**
  Ya que el robot va a actuar como sentido de la vista del usuario, grabará de forma continua imágenes del entorno que le rodea.

- **RF-CR2: Transmitir vídeo.**
  El robot tendrá la capacidad de transmitir el vídeo del entorno que capture la cámara.

- **RF-CR3: Mecanismo de transporte.**
  El robot se utilizará como dispositivo de exploración del entorno. Para ello es necesario que el robot pueda moverse en todas las direcciones, como si del propio usuario se tratara. Por ello su diseño incluye dos ruedas laterales que le permiten avanzar, retroceder y rotar sobre sí mismo.

- **RF-CR4: Recibir comandos**
  El robot tendrá un sistema de recepción de comandos a través del cual el usuario podrá controlar su movimiento.

- **RF-CR5: Comunicación a través de una red inalámbica.**
  Ya que un objetivo del proyecto es dar independecia al usuario el robot debe poder recibir los comandos a través de la red inalámbrica.

- **RF-CR6: La cámara debe apuntar en la misma dirección que la cabeza del usuario.**
  La cámara del robot hará las funciones de los ojos del usuario, por lo tanto debe tener la misma orientación respecto al cuerpo del robot que la orientación de la cabeza respecto al cuerpo del usuario.

- **RF-CR7: El robot debe llevar una fuente de alimentación integrada.**
  El robot no estará conectado a la red eléctrica por medio de un enchufe ya que esto perjudicaría a su independencia de movimientos.

- **RF-CR3: Mecanismo de rotación de la cámara.**
  El robot se utilizará como dispositivo de exploración del entorno. Para ello es necesario que la cámara pueda rotar sobre el eje horizontal. Por ello el robot tendrá un motor que permita rotar a la cámara.

3.1.2 Streaming

Se quiere conseguir la máxima sensación de inmersión para la persona que controla el robot. Para ello los requisitos principales que debe cumplir son:

- **RF-S1: La transmisión de vídeo debe tener una latencia aceptable para el control del robot por parte del usuario.**
  Si las imágenes del entorno llegan al usuario con un desfase temporal importante respecto al movimiento de la cabeza, se reducirá la sensación de inmersión y será mucho más dificil controlar el robot.

- **RF-S2: Imagen desdoblada**
  El usuario verá el vídeo con las Oculus Rift, por lo que la imagen debe estar desdoblada (formato SBS, que se eplicará en la sección de desarrollo).

- **RF-S3: Transmisión de vídeo inalámbrica**
  Al igual que en el requisito **RF-CR5**, la transmisión del streaming también debe ser inalámbrica ya que, de otro modo, se perdería la libertad dde movimiento del robot.

3.1.3 Control del Robot

Como hemos dicho anteriormente, se pretende que la cámara, soportada por el robot, sea una extensión de los ojos de usuario.Por esto es lógico que se controle con el movimiento de cabeza de la persona que esté recibiendo el vídeo. Dentro del control del robot vamos a diferenciar entre control del movimiento del robot y control de la dirección de la cámara:

- Movimiento del robot :

  RF-CTRL1: El control del robot solo depende del movimiento de la cabeza del usuario
  La interfaz de control del robot constará solo de movimientos de la cabeza del usuario ya que el trabajo está orientado a personas con discapacidad motora.

- Movimiento de la cámara:

  RF-CTRL2: La cámara debe moverse acorde con el movimiento de la cabeza del usuario.
  Este requisito es necesario tanto para el control del robot como para la sensación de inmersión. *Por ejemplo:* Es importante que si el usuario hace el movimiento de mirar hacia arriba, la imagen que le llegue sea acorde con el movimiento.

3.2 Requisitos no funcionales

- **RNF1 - Usabilidad**
  El sistema para controlar el movimiento del robot y de la cámara debe ser intuitivo, sin requerir movimientos de cabeza inusuales por parte del usuario.

- **RNF2 - Accesibilidad**
  Dado que el trabajo está orientado a personas con discapacidad motora, el control del robot debe ser accesible para las mismas, de forma que no requiera de movimientos bruscos o exagerados.

- **RNF3 - Sistema configurable**
  El sistema de movimiento y control del robot no debe ser cerrado, por el contrario debe contar con parámetros de configuración para poder adaptar el sistema a las capacidades motoras de cada usuario.

Diseño

4

Tras analizar los requisitos que debe cumplir el trabajo vamos a ver cómo se diseñará cada componente y como estarán integrados entre ellos.

Los componentes que forman el proyecto son: las gafas de realidad virtual Oculus Ridt DK2, un router(Xavi7968.), un ordenador y un dispositivo móvil con cámara integrada (robot). El diseño de los componentes, el control del robot y las conexiones se realizaran en base a los recursos y herramientas disponibles, que también se presentarán en este capítulo.

Recordemos que el objetivo es construir un dispositivo que se comunique de forma inalámbrica con el usuario. El robot debe transmir imágenes a tiempo real del entorno y recibir comandos de movimiento para que el usuario pueda explorar el espacio. Por lo tanto en términos generales, las tareas que desempeñarán los componentes son las siguientes:

- **Robot**

  Debe ser un dispositivo móvil, con una webcam integrada para poder transmitir imágenes.

  Además de la cámara el robot deberá tener integrado algún mecanismo de envío y recepción de datos.

- **Router**

  Se utilizará un router para crear una red local a través de la cual se envíen tanto las imágenes como los comandos de control del robot.

- **Oculus Rift y PC**

  Las gafas de realidad virtual reproducirán las imágenes para el usuario. Deben estar conectadas a un PC que reciba el vídeo y envíe los comandos al robot.

4.1 Diseño del robot

El robot que se utiliza en este proyecto está basado en un diseño anterior hecho por Carlos García Saura [20]. De este diseño hemos reutilizado la gran parte del diseño estructural.

En un principio también se pretendía utilizar la misma placa que el trabajo recién citado: **Arduino Yun**. Esta placa tiene la capacidad de conectarse a **internet a través de Wifi**, además tiene dos procesadores, el Atmega y el Atheros AR9331, lo que le permite de **combinar Arduino con Linux**. Esto era muy útil ya que el control de servomotores es muy sencillo con Arduino y tener un sistema operativo Linux en la placa nos permitía instalar programas de manejo de vídeo. Tras muchas pruebas se decidió no utilizar esta placa ya que el módulo Wifi daba muchos problemas y se optó por utilizar la Raspberry Pi 3, de la que se hablará más adelante.

Finalmente el robot consta de:

- Una placa base → Raspberry Pi 3.
- Dos ruedas conectadas con dos servos que permiten al robot desplazarse.
- Una cámara Logitech, conectada a la placa por USB.
- Un servo unido a la cámara.
- Una estructura de plástico impresa con la impresora 3D del Club de Robótica de la EPS.

A continuación se explicarán en detalle cada uno de los componentes del robot.

4.1.1 Raspberry Pi 3 model B

La Raspberry Pi 3 model B C.1 es un ordenador de placa reducida que lleva un sistema operativo Linux. Su procesador es un ARM Cortex A53 de cuatro núcleos a 1.2GHz de 64 bits. Tiene 1 GB de memoria RAM, 4 puertos USB, 40 pins GPIO, puerto HDMI, Ethernet y entrada para MicroSD. Además tiene Wifi 802.11n integrado y bluetooth 4.1.

Tras rechazar el Arduino Yun como placa para el proyecto, la Raspberry Pi 3 quedó como mejor candidata ya que cumplía todos los requisitos necesarios. A saber:

- **Módulo Wifi (802.11n Wireless LAN)**: Imprescindible para la transmisión.
- **S.O Linux**: Nos permite instalar programas de manejo y transmisión de vídeo, como la librería mjpeg-streamer que se usará en este proyecto.
- **Puerto USB**: Para conectar la cámara y la fuente de alimentación.
- **GPIO**: Necesario para conectar los servomotores.

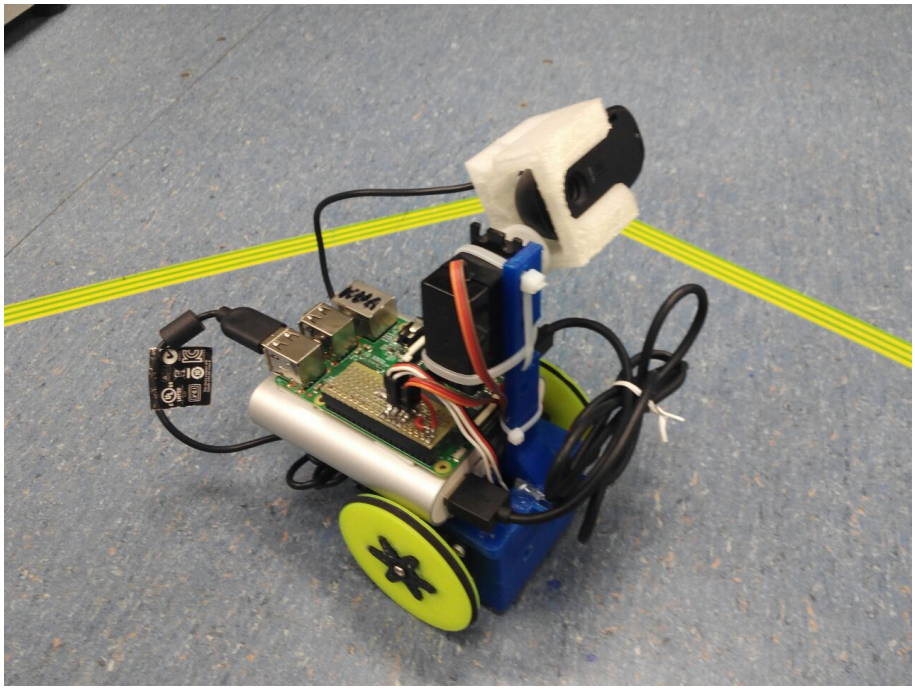


Figura 4.1: Robot de este trabajo con la Raspberry Pi 3

4.1.2 Servo motores

El robot debe tener la capacidad de desplazarse, por lo que se diseñó con dos ruedas laterales que son controladas a través de dos servomotores de rotación continua. Otro requisito era que la cámara debía estar en todo momento orientada respecto al robot hacia la misma dirección que la cabeza del usuario está orientada respecto a su cuerpo **RF-CR6**. Para poder cumplir este requisito se añadirá un servomotor al robot que esté unido a la cámara y pueda rotarla respecto al eje horizontal.

Antes de mostrar cómo se van a conectar los servomotores con las ruedas y la cámara y cómo reciben los comandos, se va a explicar brevemente qué es un servomotor y cómo funciona:

Un servomotor es un dispositivo similar a un motor de corriente continua que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación, y mantenerse estable en dicha posición. Una de las características más importantes de estos dispositivos es que puede controlarse tanto en posición como en velocidad.

Los servos constan de:

- Un motor eléctrico
- Una caja reductora
- Un circuito de control

El sistema que utilizan los servomotores para controlar la velocidad y la posición de los motores eléctricos es la modulación por ancho de pulso (PWM).

PWM Este sistema consiste en generar una onda cuadrada en la que se varía el tiempo que el pulso está a nivel alto, manteniendo el mismo período (normalmente), con el objetivo de modificar la posición según se desee.

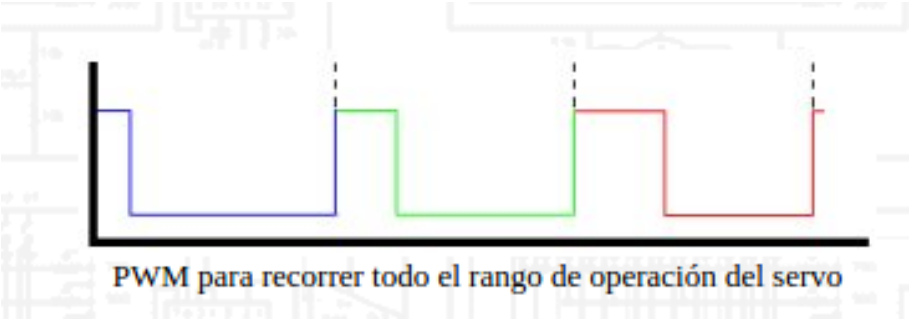


Figura 4.2: Onda cuadrada que representa los pulsos que se envían a los servomotores

Como hemos dicho, el control de la posición y la velocidad es un atributo característico de este tipo de motores y por tanto se va a explicar brevemente cómo funciona.
Para que un servo se mantenga en la misma posición durante un cierto tiempo, es necesario enviarle continuamente el pulso correspondiente.
El control de la velocidad se consigue alimentando el motor con una señal de pulsos con la frecuencia suficiente para que el motor no note las variaciones y haga un giro constante, ya que variando el porcentaje de tiempo de la señal rectangular en alta, y en baja, variamos la potencia que le entregamos al motor, con lo que podemos controlar la velocidad de giro.

En el robot utilizamos 3 servomotores. Dos de ellos se utilizan para el movimiento de las ruedas y son servomotores de corriente continua y el tercero es un servomotor paso a paso que moverá la cámara permitiendo al usuario mirar hacia arriba y hacia abajo. Los tres servos están controlados por el movimiento de la cabeza del usuario, que obtenemos gracias a las Oculus Rift.

4.1.3 Cámara

La cámara utilizada es una webcam USB. El proyecto también se podría haber hecho con una RaspiCam o con una cámara Wifi. Se optó por la cámara Logitech ya que era la que tenía disponible por lo que hacía el proyecto más económico y la placa ya contaba con tarjeta de red, por lo que no era necesario que la cámara también contara con módulo wifi.

A continuación se muestra un esquema 4.3 de la conexión entre los componentes que acabamos de explicar.
La placa gris que aparece en el esquema sirve para unificar las entradas y salidas de los servomotores, de esta forma un solo pin de la raspberry controlará los pulsos que se envían a los dos servomotores de las ruedas del robot.
En el robot estas conexiones se han hecho soldando los cables a una placa mucho más reducida que la que aparece en el dibujo.

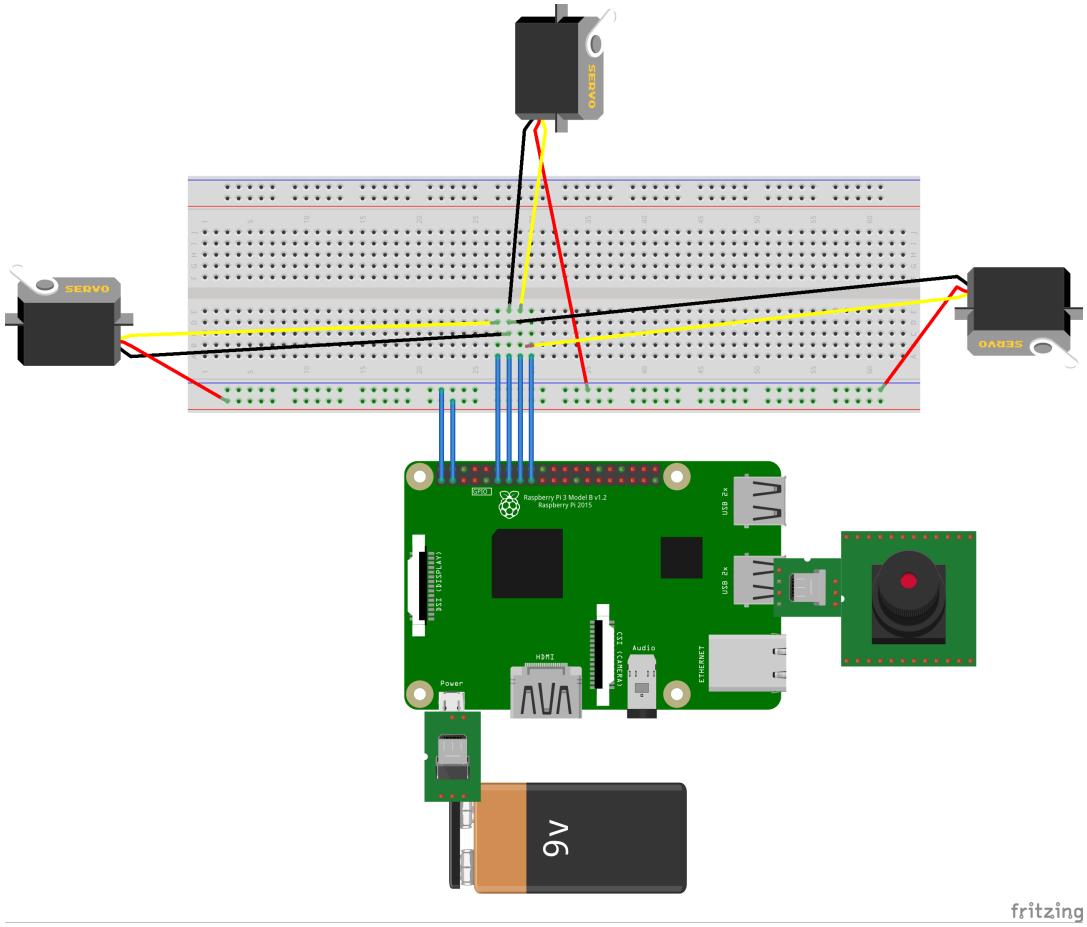


Figura 4.3: Esquema conexiones servos, cámara, Raspberry Pi 3

4.2 Router

Para la comunicación entre el robot y el ordenador se necesita una red local, para este trabajo la hemos creado haciendo uso de un router inalámbrico Xavi7968. El router crea una red Wifi con el nombre de WifiRaspi3 y sin acceso a internet.

Cuando un dispositivo se conecta a una red, el router le asigna de forma automática una dirección IP que le identificará respecto a los otros dispositivos conectados a la misma red.
En este trabajo es necesario conocer la dirección IP de la Raspberry Pi 3 y del ordenador, y además necesitamos que siempre sea la misma, de esta forma será sencillo automatizar la conexión entre ambos.
Las direcciones IP las genera de forma automática un servidor DHPC (Dynamic Host Configuration Protocol), pero el router nos permite configurar dicho servidor y asignar a una dirección MAC una dirección IP. En términos más sencillos, podemos "reservar" una o varias direcciones IP de la red local y emparejar estas IPs a los dispositivos pertinentes.
Como podemos ver en la imagen A.1 que se encuentra en el apéndice, se ha configurado la tabla DHCP del router para asignar siempre la misma IP a la Raspberry y al equipo que utilizamos para conectarnos.

Una vez configurada la tabla DHPC, configuramos la Raspberry Pi 3 para conectarse automáticamente al router.
Esto se hace modificando el archivo `/etc/wpa_supplicant/wpa_supplicant.conf`,

donde se ha añadido el nombre y la clave de la red creada por el router. La imagen puede verse en los apéndices A.2.

4.3 Oculus Rift

Las Oculus Rift son unas gafas de realidad virtual creadas por Palmer Luckey. Ha habido varias versiones de estas gafas. La primera fue la DK1 (Develoment Kit 1), enfocada para programadores, que salió en abril de 2013. Esta versión se retiró del mercado poco antes de la presentación del segundo kit de desarrollo (DK2) en marzo de 2014. Finalmente, en 2016 salió la versión para consumidor.
Para este proyecto se utiliza la versión **DK2**, que se muestran en la imagen 4.4, ya que eran las gafas que teníamos disponibles.

Las gafas son un componente especialmente importante en este trabajo, ya que son el dispositivo que conecta al usuario con el entorno y el que transmite la información del robot al usuario y del usuario al robot.
Por ello las funciones clave de las Oculus Rift para el proyecto son:

- Detección de la orientación: Las gafas llevan integradas sensores que permiten al ordenador conocer su orientación y transmitirla al robot.
- Sensación de inmersión (3D): dada la tecnología de las lentes de las Oculus Rift, es posible crear en el usuario una sensación de inmersión en el espacio en el que se encuentra el robot.

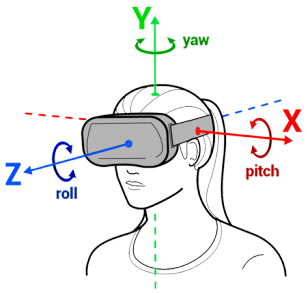
4.3.1 Detección de la orientación

Las Oculus Rift son capaces de detectar la **posición y orientación de la cabeza** del usuario.

Para la **posición**, las gafas llevan un conjunto de emisores infrarrojos en su superficie. La información enviada por dichos emisores es recogida por un sensor de infrarrojos que se asemeja a una cámara.
Para nuestro trabajo la posición no es relevante ya que, al estar orientado para personas con discapacidad motora, se espera que el usuario se encuentre más o menos en la misma posición durante el uso del dispositivo.Por otra parte el usuario puede estar algo inclinado o incluso tumbado ya que, al no leer los sensores infrarrojos, el movimiento del robot no depende de la posición del usuario.

Por el contrario, conocer la **orientación** de la cabeza es esencial en este proyecto ya que el usuario enviará comandos al robot rotando la cabeza en el sentido que convenga en cada momento.
Esta orientación se lee respecto a los ejes X, Y, y Z, como se puede ver en la figura 4.4 y se detecta gracias a los tres sensores que las Oculus Rift llevan integrados (giroscopio, acelerómetro y magnetómetro).

Con el giroscopio y el acelerómetro se calcula la orientación respecto a los ejes X y Z, mientras que el magnetómetro manda información del eje Y.
Combinando la información de estos sensores a través de un proceso conocido como fusión de sensores se determina la orientación de la cabeza del usuario en el mundo real, y se sincroniza con la perspectiva virtual del usuario en tiempo real.
En este trabajo los giros de la cámara serán respecto a los ejes X e Y por lo que tampoco se recogerá información de la orientación de la cabeza respecto al eje Z.



(a) Ejes de giro

(b) Oculus Rift DK2. Gafas de realidad virtual que se utilizan para este trabajo

Figura 4.4

Se ha implementado un programa en phyton que recoge esa información y la traduce en comandos sencillos para mandarselos a la Raspberry. Para obtener esa información hacemos uso de la API de python (ovr 4.3.3) que se comunica con la librería de Oculus (LibOvr).

4.3.2 Sensación de inmersión 3D

Una de las características más importantes de las Oculus Rift es su capacidad de reproducir un entorno virtual de tal forma que el usuario se sienta totalmente parte de él.

Esto lo hace usando un par de pantallas (una para cada ojo) que dividen la imagen en dos (Side by side - SBS). En cada pantalla hay un conjunto de lentes que modifican la imagen para crear el fenómeno de la estereopsis por el cual, a partir de dos imágenes ligeramente diferentes proyectadas en la retina de cada ojo, el cerebro es capaz de recomponer una tridimensional.

4.3.3 OVR

Para desarrollar una aplicación integrada con las Oculus Rift utilizaremos la API OVR de python, que se encuentra en `https://github.com/cmbruns/pyovr`. Esta API está basada en otra API escrita en C que nos permite utilizar la librería LibOVR. Esta librería es la parte fundamental del SDK de Oculus.

Todo software que utilice OVR debe tener al menos 3 fases:

- Inicialización.
- Bucle principal.
- Liberación de recursos y fin de proceso.

Inicialización

Para inicializar la librería se utiliza la función `ovr.initialize(None)`. Esta función debe llamarse al principio de cualquier aplicación que utilice OVR.

Una vez inicializada la librería es necesario llamar a `ovr.create()`, que se comunica con el sistema operativo y crea el entorno necesario para gestionar las Oculus. Esta función devuelve dos valores: session (la sesión ovr) y luid (locally unique identifier).

Bucle principal

En el caso de este trabajo, el bucle principal deberá leer la orientación de las gafas y enviarla al robot.

Para obtener la orientación es necesario recoger la información de los sensores de las Oculus Rift. Para ello OVR proporciona la función `ovr.getTrackingState()`, que devuelve una estructura *TrackingState*. En esta estructura se encuentra el campo *HeadPose*. Leyendo *HeadPose.ThePose.Orientation* se encuentra la información necesaria sobre la orientación de las Oculus respecto a los ejes x e y.

Liberación de recursos y fin de proceso.

Cuando finaliza la aplicación se deberá llamar a las funciones `ovr.destroy(session)` y `ovr.shutdown()`. Con estas funciones liberamos los recursos utilizados en la aplicación y apagamos la API.

4.4 Diseño de conexiones entre módulos

La arquitectura del proyecto es muy sencilla, como podemos observar en la siguiente imagen 4.5:

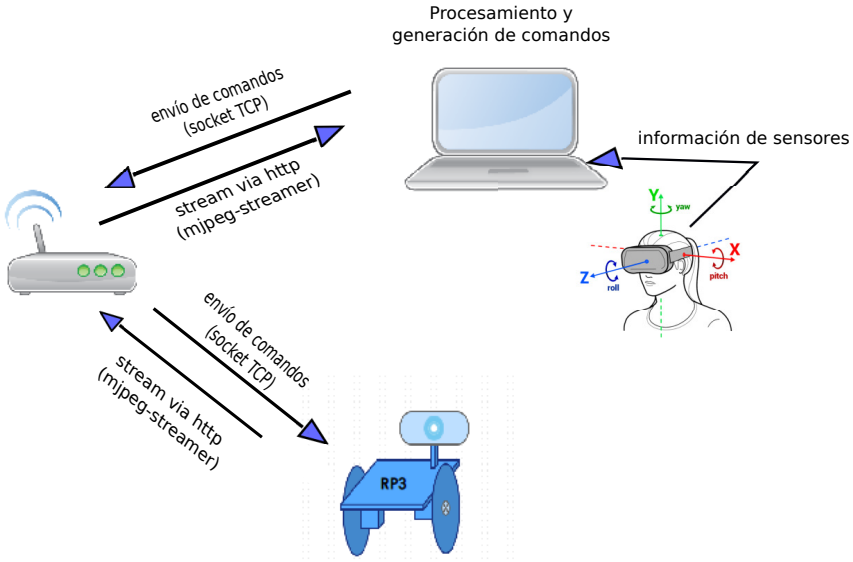


Figura 4.5: Esquema conexiones

Todo el sistema, excepto las Oculus, se conecta a través de la red local creada por el router.
Para hablar de la conexión de las Oculus hay que especificar que nos referimos a la conexión de las gafas y la conexión de la cámara receptora de infrarrojos.
La cámara receptora de infrarrojos se conecta con el ordenador a través de un puerto USB y las gafas se conectan con la tarjeta gráfica a través de HDMI.

El resto de dispositivos, es decir, el ordenador y el robot, se encuentra dentro de la red local WifiRaspi3.De esta forma el ordenador envía comandos de movimiento al router y el router al robot. Por otra parte la cámara (conectada por USB a las Raspberry Pi 3) transmite el vídeo al robot y este lo manda en tiempo real al router, el router lo envía al ordenador de forma que el usuario podrá visualizarlo con las gafas de realidad virtual.

4.5 Esquema funcional

A continuación se presenta un esquema funcional de todo el proyecto, en el que se puede ver cómo interacciona cada componente con los demás.

El esquema muestra un **ciclo retroalimentado**, ya que la información que manda el robot al usuario condiciona los comandos de movimiento que mandará el usuario al robot y viceversa.

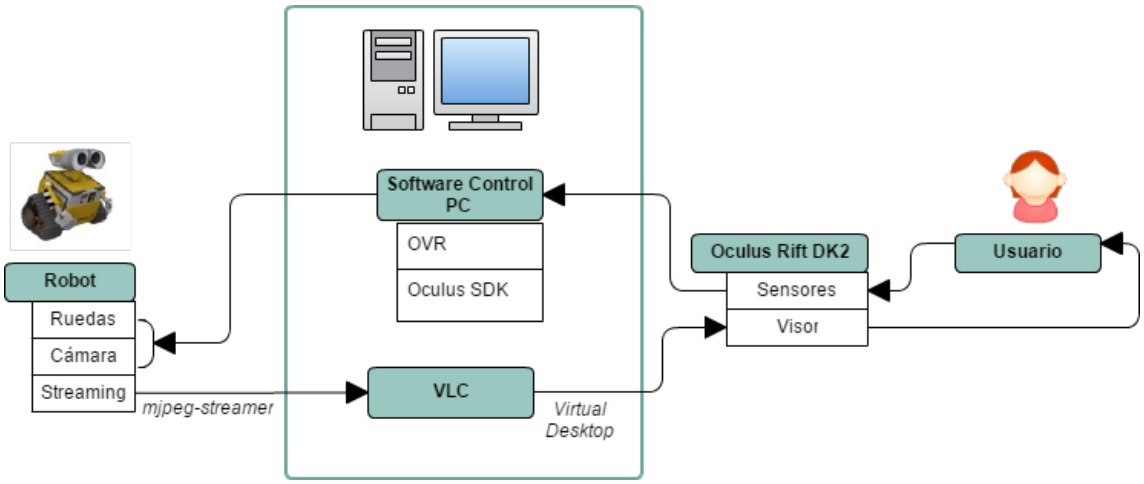


Figura 4.6: Esquema funcional

Empezando el ciclo desde el **usuario**, vemos cómo el usuario le manda información a los sensores, esto es, cuando el usario mueve la cabeza, los sensores de las **Oculus** perciben el cambio de orientación y le mandan los datos al PC.

El PC recoge los datos a través del SDK de Oculus. Estos datos son interpretados en un **software de control** utilizando la API OVR. Una vez interpretados los datos se envían los comandos convenientes a los motores del **robot**. Estos comandos llegarán a los motores de las ruedas o al motor de la cámara, dependiendo de la nueva orientación de la cabeza del usuario.

A la vez que ocurre este proceso el robot está enviando un vídeo del entorno en tiempo real al PC. Esta transmisión se hace a través de la librería mjpeg-streamer y es recogida en el PC por el reproductor de vídeo **VLC**.

Para que el vídeo llegue al visor de las Oculus Rift se utiliza Virtual Desktop, como ya se dijo en la sección anterior. De esta forma el usuario podrá visualizar lo que ve el robot a través de la Oculus y mover el robot hacia donde desee.

5 DESARROLLO E IMPLEMENTACIÓN

Una vez definidos los objetivos y requisitos que debe cumplir el trabajo y haber diseñado el sistema y las conexiones, se ha desarrollado el sistema basado en todo lo anterior.
En esta sección se explicará cómo hemos desarrollado e integrado las distintas funcionalidades para conseguir el proyecto final. Ya que el proyecto es tan amplio e integra tantas tecnologías,para clarificar la explicación se ha dividio el desarrollo en tres bloques:

- Construcción del Robot
- Streaming
- Control del Robot

Cada bloque se desarrollará a lo largo del capítulo pero, para tener una idea global, primero se van a resumir brevemente.

La **construcción del robot** se ha hecho acorde con el diseño y los requisitos del mismo, es decir, con dos ruedas que le permitan transportarse, un soporte para la Raspberry Pi 3 y un soporte para la cámara.
Este diseño se basa en un diseño de otro trabajo de un alumno de la UAM, Carlos García Saura [20], añadiendo para este proyecto los accesorios necesarios.

Para conseguir **transmitir a las Oculus el vídeo en tiempo real** es necesario que la Raspberry recoja el vídeo de la cámara y lo transmita al ordenador. A su vez el ordenador recojerá el vídeo, lo transformará a formato SBS y lo reproducirá en las Oculus Rift.

El último bloque es el **control del robot**. El primer paso para poder enviar comandos de control es leer la señal de los sensores de las Oculus Rift (giroscopio, acelerómetro y magnetómetro) para saber la orientación de la cabeza del usuario.
Una vez que obtenemos dicha información se procesa y transforma en datos útiles para el robot y se transmite a las Raspberry Pi con la mínima latencia posible.
Cuando el robot recibe los datos los convierte en instrucciones para mover los servomotores, es decir, en pulsos que le indican un cambio de posición.

5.1 Construcción del Robot

Como se ha dicho en la sección de diseño, el robot está construido basándose en el diseño de un robot de un trabajo anterior [20].
Para este proyecto se han añadido dos componentes adicionales:

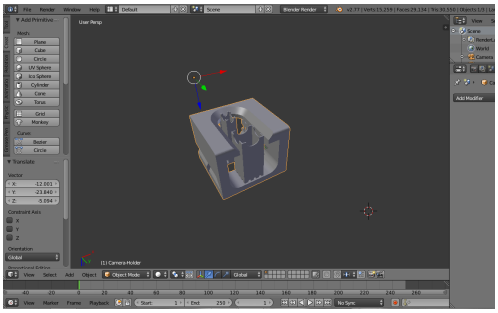
- **Soporte de la cámara:** este soporte debe servir como pinza de agarre de la cámara y a su vez debe estar unida al servomotor que mueve la cámara.
- **Batería portátil:** Ya que el robot debe ser inalámbrico, de acuerdo con los requisitos del proyecto, en su estructura hay cabida para una batería portatil que alimenta la Raspberry Pi 3 y a través de ésta, la cámara y los tres servomotores.

5.1.1 Soporte de la cámara

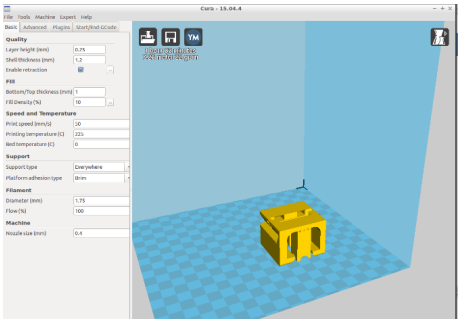
El soporte de la cámara se ha construido en base a que la cámara debía encontrarse a una altura algo elevada sobre la base del robot para poder rotar sin encontrarse con obstáculos y para lograr un mayor ángulo de visión. Otro requisito que tiene que cumplir el soporte es el de dar lugar a que se coloque el servomotor que controla el movimiento de la cámara.
En base a estas dos condiciones se ha ideado un soporte con dos piezas: Un soporte vertical y una pinza de agarre.

- Soporte vertical → En esta pieza se coloca el servomotor que controla el movimiento de la cámara.
- Pinza de agarre → Se utiliza para unir la cámara con el servomotor que a mueve.

Ambas piezas han sido impresas en plástico con la impresora 3D del Club de Robótica de la Universidad Autónoma de Madrid.
Para realizar el diseño 3D de la pinza de agarre se ha utilizado el programa de diseño 3D Blender.
En este programa se diseña gráficamente la pieza y genera una archivo `.stl`, tal y como se muestra en la imagen 5.1(b).



(a) Diseño 3D de la pinza de agarre en Blender



(b) Imagen de la configuración del Cura para enviar la Pieza de agarre a la impresora 3D

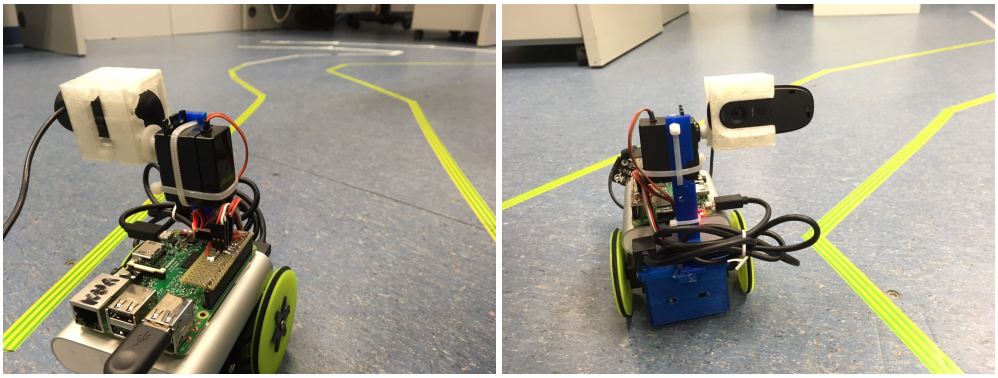
Figura 5.1: Vista de la pinza de agarre en Blender y en Cura

Una vez creado el fichero `.stl` utiliza Cura [1], un programa que nos permite seleccionar los parámetros de configuración deseados para la impresión de la pieza, como

[1] Página de descarga de Cura: https://ultimaker.com/en/products/cura-software

por ejemplo la densidad o el tamaño. Una vez elegida la configuración, Cura genera el archivo `.gcode`. Este tipo de archivos son los que lee la impresora 3D y consiste en una serie de instrucciones de movimiento para los servomotores que controlan el estrusor y el ventilador de la impresora.
En la imagen 5.1(a) podemos ver la interfaz de Cura, la base azul del centro representa la base de impresión de la impresora. A la izquierda se pueden ver algunos de los parámetros de configuración.

Una vez imprimidas las piezas ya se puede construir el robot. El resultado se muestra en la imagen 5.2:



(a) Vista trasera del robot (b) Vista frontal del robot

Figura 5.2: Imágenes del robot una vez montado

5.2 Streaming

El segundo bloque es el streaming, es decir, la transmisión del vídeo que captura la cámara al ordenador y su posterior reproducción en las Oculus Rift.
La Raspberry Pi3 se encargará de recoger el vídeo de la cámara y transformarlo en paquetes de datos IP para poder transmitirlos a una dirección de la red local.
El ordenador va recogiendo el streaming de la dirección donde lo envía la Raspberry Pi 3 y lo reproduce en las Oculus.

5.2.1 Raspberry Pi3 - Transmisión de vídeo

La Raspberry Pi, tiene conectada por usb la cámara que se utiliza para capturar el vídeo.
Para recoger el vídeo y mandarlo en tiempo real utilizamos la aplicación de software libre mjpeg-streamer, disponible en github. `https://github.com/jacksonliam/mjpg-streamer`

Mjpeg-streamer es una aplicación en linea de comandos que permite crear un servidor, para obtener frames JPG desde una cámara compatible con la aplicación y transmitirlos como M-JPEG mediante protocolo HTTP. Está diseñada para dispositivos con recursos limitados, en téminos de CPU y RAM.
Soporta la compresión por hardware (GPU) de la cámara, en nuestro caso H.264 Advanced Video Coding (AVC) Standard, que es la compresión utilizada por la Webcam Logitech, lo cual permite reducir drásticamente el uso de la CPU de este servidor, haciendo está aplicación un servicio ligero.
Esta característica hace de esta aplicación una buena candidata para el objetivo que

perseguimos en este trabajo, ya que la compresión y transmisión del streaming se hace a tiempo real.

En la imagen 5.3 se muestra el uso de recursos de la aplicación mjpeg-streamer.

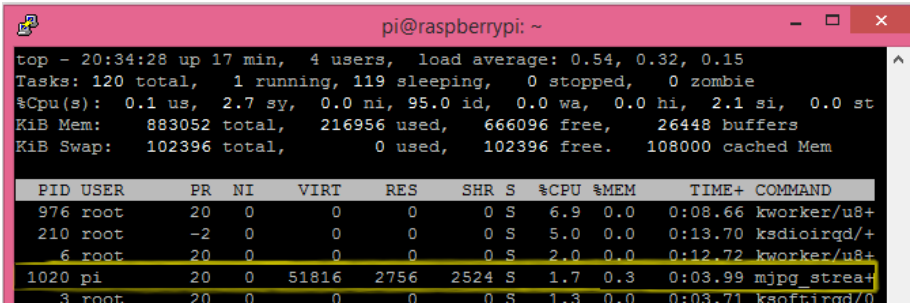


Figura 5.3: Uso de CPU del programa Mjpeg-Streamer por el que se hae el streaming de vídeo

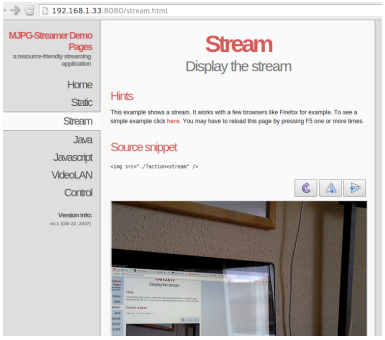
La aplicación arranca en la Raspberry Pi cuando ejecutamos el siguente comando: `./mjpg_streamer -i "./input_uvc.so -d /dev/video0" -o "./output_http.so -w ./www"`.
Para entender este comando hay que explicar primero el funcionamiento de mjpeg-streamer. La aplicación se basa en unos plugins de entrada y salida. Es decir, un plugin (de entrada) copia las imágenes JPEG a un directorio de acceso global, mientras que otro plugin (de salida) procesa las imágenes y las emite de acuerdo a los estándares MPG existentes.
En el comando que arranca la aplicación estamos especificando:

- **-i "./input_uvc.so**: Este es el plugin de entrada, que captura los frames JPG de la cámara conectada.
- **-o "./output_http.so -w ./www"**: El plugin de salida (`./output_http.so`), es un servidor web HTTP 1.0, que emite el vídeo en los estándares M-JPEG. El directorio de la aplicación web de mjpeg-streamer se especifica con la opción `-w ./www`
- **mjpg_streamer**: instrucción que transmite los frames desde el plugin de entrada al de salida.
- **-d /dev/video0**: especifica la cámara que se va a utilizar, ya que la Raspberry Pi 3 tiene dos puertos usb.

En el momento en que se ejecuta el comando, la Raspberry empezará a recoger el vídeo de la cámara y transmitirlo en tiempo real a la siguiente url: `http://192.168.1.33:8080/stream.html` y muestra información sobre el streaming por terminal.
En la imagen 5.4 vemos un ejemplo de dicha información y la dirección de la red local donde la Raspberry Pi está transmitiendo el vídeo.

```
MJPG Streamer Version.: 2.0
 i: Using V4L2 device.: /dev/video0
 i: Desired Resolution: 640 x 480
 i: Frames Per Second.: -1
 i: Format............: JPEG
 i: TV-Norm...........: DEFAULT
 o: www-folder-path...: ./www/
 o: HTTP TCP port.....: 8080
 o: username:password.: disabled
 o: commands.........: enabled
```

(a) Ejecución del comando para iniciar el streaming



(b) Página html donde se recibe el streaming a tiempo real desde la Raspberry

Figura 5.4: Ejecución de mjpeg-streamer

5.2.2 Ordenador - Recepción de vídeo

Una vez el robot es capaz de transmitir vídeo a tiempo real a una dirección de la red local, solo falta desarrollar las funciones del ordenador.
El PC debe capturar el vídeo de la red local y reproducirlo en las Oculus Rift.
El ordenador, conectado a la red local a través del router,accede a la IP donde mjpeg-streamer está retransmitiendo el vídeo y lo recoge a través del reproductor de vídeo VLC [2].

VLC media player es un reproductor multimedia libre y de código abierto. Tiene dos características que lo hacen idóneo para este proyecto: Es capaz de capturar y reproducir vídeo de una dirección de red y puede dividir la imagen del vídeo en dos pantallas.

Por lo tanto se configura VLC para que el medio de reproducción sea la ubicación de red donde se está transmitiendo el vídeo del robot `http://192.168.1.33:8080/?action=stream`.
Posteriormente dividimos el vídeo en dos pantallas exactamente iguales para poder verlo desde las Oculus en modo SBS (Side by Side).
Ambas configuraciones y el resultado pueden verse en la figura B.1 del apéndice B.

Una vez capturado el vídeo en el ordenador solo queda visualizarlo en las Oculus Rift.
Para ello hemos instalado en el PC el Virtual Desktop [3].

Virtual Desktop es una aplicación desarrollada por Oculus Rift y HTC Vive para utilizar las Oculus Rift en Windows.
Permite ver el escritorio desde las Oculus, pudiendo configurar efectos como ver las imagénes SBS o cambiar el entorno en el que te encuentras (el espacio, una sala de cine...)
También te permite ver vídeos descargados en el ordenador y reproducir vídeos 360.
La funcionalidad que es interesante para este trabajo es la de ver la pantalla del ordenador desde las Oculus con SBS, para visualizar el vídeo que se está reproduciendo en VLC a tiempo real.
De esta forma, si ponemos en vídeo en pantalla completa en el PC y configuramos Virtual Desktop correctamente, el vídeo se reproducirá en la Oculus Rift a tiempo real.

[2] Página oficial de VLC: http://www.videolan.org/vlc/
[3] Página oficila de Virtual Desktop: http://store.steampowered.com/app/382110/

5.3 Control del Robot

El último bloque de desarrollo que queda por implementar es el control del robot. Por control del robot se entiende el control del movimiento de los tres servos que lo componen. Dos para las ruedas laterales que permiten que el robot se transporte y uno para la cámara, que permite que esta rote respecto al eje horizontal.
Estos movimientos se basarán en el movimiento de las Oculus, que llevará puestas el usuario.
Por lo tanto el control del robot se divide en cuatro acciones:

- Control del robot por parte del usuario
- Recogida y procesamiento de la información que mandan las Oculus al ordenador → Esto se hará mediante la utilización de la API de Python `OVR`.
- Envío de la información procesada a la Raspberry Pi 3 → Para ello el ordenador abrirá dos sockets a través de los cuales mandará los datos al robot
- Movimiento de los servomotores

5.3.1 Control del Robot por parte del usuario

Como se especifica al inicio de este trabajo, el robot estará controlado exclusivamente por el movimiento de cabeza del usuario.

Para ello se debía idear una forma cómoda e intuitiva de mandar instrucciones al robot.

Diferenciamos dos estados de movimiento del robot:

- Exploración→ En este estado el robot no avanza, solo explora lo que tiene a su alrededor. Esto lo hace moviendo la cámara o rotando sobre sí mismo.
- Transporte → El robot avanza hacia delante. Al poder girar sobre sí mismo en la fase de **exploración**, es suficiente con este movimiento.

Finalmente se ha implementado de la siguiente forma:

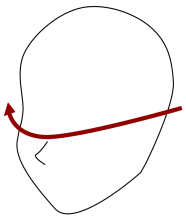
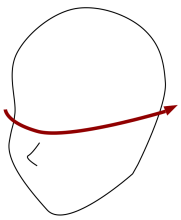
Fase de exploración: Si el usuario gira la cabeza hacia la **derecha** el robot girará sobre si mismo hacia la **derecha**, sin avanzar ni retroceder, hasta que el usuario vuelva a mirar hacia el centro	**Fase de exploración:** Si el usuario gira la cabeza hacia la **izquierda** el robot girará sobre si mismo hacia la **izquierda**, sin avanzar ni retroceder, hasta que el usuario vuelva a mirar hacia el centro
	

Tabla 5.1: Rotaciones hacia la derecha y hacia la izquierda

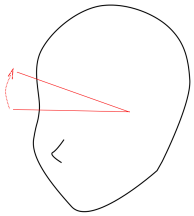
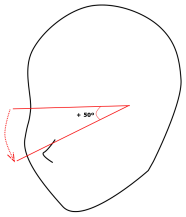
Fase de exploración: Si el usuario gira la cabeza hacia **arriba** sobre el eje horizontal la cámara rotará hacia arriba sin avanzar ni retroceder	**Fase de transporte:** Si el usuario gira la cabeza hacia **abajo** con un ángulo menor que -30^{o} tal y como se indica en la figura, el robot avanzará hacia delante en linea recta hasta que el usuario vuelva a subir la cabeza.
	

Tabla 5.2: Rotaciones verticales

Como viene explicado en la imagen 5.1, si el usuario mira hacia los lados el robot rotará sobre sí mismo sin moverse del sitio, hacia la dirección que señalen las Oculus Rift.(**exploración**)
Si el usuario mueve la cabeza hacia arriba o hacia abajo sin llegar a menos de -30^{o}, la cámara se moverá como si fuera la cabeza del usuario.(**exploración**)
En el caso de que el usuario oriente la cabeza hacia abajo, con un ángulo menor de -30^{o} respecto al eje Y, la cámara se orientará horizontalmente (0^{o} respecto al eje Y) y el robot avanzará hasta que el usuario cambie la posición de la cabeza.(**transporte**)

5.3.2 Recogida y procesamiento de la información que mandan las Oculus al ordenador

Para recoger desde el ordenador la información que mandan las Oculus utilizamos la API de Oculus (`OVR`), explicada en el capítulo de Diseño 4.3.3.
Con las funciones proporcionadas por OVR se puede conocer la orientación de la cabeza del usuario respecto a los ejes x e y.
La información obtenida sobre la orientación respecto al eje y servirá para mover los servos que hay en las ruedas del robot, mientras que la información de x servirá para mover la cámara.
Los datos referentes a la posición de las Oculus se convierten grados, lo que facilita el control posterior de la posición de los servomotores.
Esto se hace de la siguente forma:

La API `OVR` nos proporciona una función ovr.getTrackingState() que nos devuelve un cuaternión que representa la orientación de las Oculus Rift en un el espacio tridimensional. La transformación del cuaternión al ángulo respecto a los ejes es la siguiente:

$$pitch = arctan(2 * (x * w + y * z), 1 - 2 * (z * z + w * w))$$

$$yaw = arcsin(2 * (x * z - w * y))$$

Siendo x, y, z, y w las componentes del cuaternión, $pitch$ en ángulo respecto al eje horizontal y yaw el ángulo respecto al eje vertical.
El código de esta parte se encuentra en los apéndices, en las imágenes E.1E.2

El ordenador crea un servidor y abre dos sockets distintos, uno para la información que manda al servomotor de la cámara y otro para los servomotores de las ruedas.
De esta forma manejamos los dos movimientos de forma independiente.

Envío de la información procesada a la Raspberry Pi 3 Para el control del movimiento del robot es necesario que el ordenador le mande los datos referentes a la posición de las Oculus de forma automática, inmediatamente después de procesarlos.
Con este fin, desde el mismo programa donde se recibe y procesan los datos de las Oculus, el ordenador abre dos sockets, y se queda a la espera de que el robot se conecte.
De esta forma se crea un sistema cliente-servidor en el cual el PC actúa como servidor y la Raspberry Pi como cliente.
Recordemos que el ordenador y la Raspberry Pi 3 están dentro de la misma red local (WifiRaspi3), y que ambos tenían IP fija, lo que hace posible automatizar la conexión entre ambos dispositivos.

La apertura de sockets por parte del ordenador se hace de la siguiente forma:

```
host = "192.168.1.35"    #IP asignada al ordenador
port = 4446        #movimiento de la camara
port2 = 4447       #movimiento de las ruedas

s = socket(AF_INET, SOCK_STREAM)
s2 = socket(AF_INET, SOCK_STREAM)

s.bind((host,port))
s2.bind((host,port2))

s.listen(5)
s2.listen(5)

q,addr = s.accept()
q2,addr2 = s2.accept()
```

De esta forma, cada vez que se recibe y procesan datos referentes a la posición de las Oculus, se dividen en información para la cámara e información para las ruedas y se ejecutan las instrucciones `q2.send(data)` y `q.send(data)` para mandar los datos.

Recepción de los datos en el robot La Raspberry tiene dos programas (uno para las ruedas y otro para la cámara) que están escuchando continuamente lo que manda el ordenador y en cuanto recibe el comando lo manda a los servos.
Para ello, al iniciarse cada programa, se conectan al socket correspondiente

```
host = "192.168.1.35"
port=4447
s=socket(AF_INET, SOCK_STREAM)
s.connect((host,port))
```

Y se ponen a escuchar hasta que les llegan los datos.

Una vez que tiene los datos los transforma en comandos válidos para los servomotores, tal y como se explica en el siguiente apartado, les manda la instrucción y vuelve a escuchar.

5.3.3 Movimiento de los servomotores

Como ya se indicó en la sección de Diseño, el movimiento de los servos se controla con una señal PWM (Pulse Width Modulation).

Por ello utilizamos la librería RPi.GPIO, que ofrece funciones para PWM.
Hay dos parámetros principales para determinar el pulso que se envía al motor:

- **Frecuencia:** veces por segundo que se genera el pulso.
- **Duty cycle:** indica el porcentaje de tiempo que dura la señal cuadrada en relación con el periodo.

  $$DutyCycle = \frac{PW}{T} \cdot 100$$

  Donde PW indica el ancho de pulso (pulse width), es decir, el tiempo que la onda se encuentra en estado alto durante un periodo, y T es el periodo.

El parámetro Duty Cycle es el que la Raspberry le envía a los servomotores para modificar su posición. Si nos fijamos en la fórmula, las variables que debemos calcular para poder enviar el Duty Cycle son PW, es decir, el ancho de pulso y T.
Para controlar los servomotores del robot se fija una frecuencia de 50 Hz, por tanto queda un periodo $T = \frac{1}{50} = 20ms$. En las especificaciones de los servos se indica que el rango de PW es 0.5 ms ~ 2.5 ms y una vez que sepamos qué ancho de pulso queremos mandar ya podemos calcular el Duty Cycle adecuado para cada acción.
En la imagen 5.5 se muestra cómo varía la posición de los motores según el ancho de pulso que se le envíe.

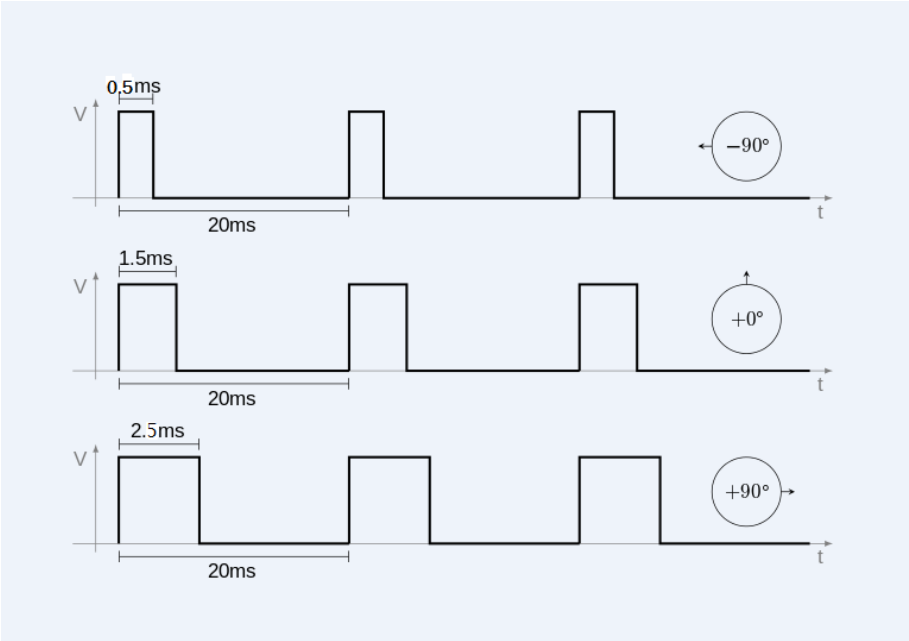


Figura 5.5: Relación entre el PW y la posición de los servomotores

Recordemos que estos valores son orientativos, pudiendo variar ligeramente según el servomotor.

Una vez entendida la relación entre el ángulo de orientación del servomotor y el ancho de pulso que recibe, se podrá entender cómo se controlan los servomotores del robot.
El control para los servomotores de las ruedas del robot y el control del servomotor de la cámara son distintos. Esto es porque los primeros son servomotores de rotación continua mientras que el segundo es un servomotor paso a paso.

Control de los motores de las ruedas del robot Los motores que mueven las ruedas del robot son dos servomotores de rotación continua con un rango de rotación de 360º. Un motor de rotación continua está modificado para que actúe como si siempre estuviera orientado hacia 0º. De esta forma el ancho de pulso que recibe el motor no indica la posición en la que debe colocarse si no la dirección y velocidad a la que debe rotar.
Para clarificar esto se presenta el siguiente ejemplo: Si el servomotor de rotación continua recibe un ancho de pulso de 0.5 ms, como el servomotor cree estar orientado hacia 0º, girará hacia la izquierda de forma continua intentando alcanzar la orientación correspondiente a -90º y no parará de girar hasta que recibir otro ancho de pulso. De esta forma se controla la dirección.
La velocidad es algo más sutil. Si el servomotor está en la posición 0 y debe moverse hasta conseguir -90º, girará muy rápido, ya que la idea es que tarde lo mismo en llegar a los -90º que a los -45º o a cualquier otra posición. Por lo tanto mandando un PW de 0.5 ms el servomotor de rotación continua girará hacia la izquierda más rápido que si recibe un PW de 1 ms.

Por lo tanto si se envía un ancho de pulso cercano a 1.5 ms, el motor girará muy despacio. Por el contrario, si el ancho de pulso es cercano a 0.5 o 2.5 ms, el motor se moverá a mayor velocidad.
Para controlar nuestro robot se ha fijado una velocidad constante de giro en base a pruebas experimentales que se explicarán en la sección de Pruebas y resultados.
Para girar el robot hacia la derecha el usuario enviará la señal de "rotar hacia la derecha" y el robot empezará a rotar en esa dirección hasta recibir la señal de "parar".

El código para el control de las ruedas del robot se encuentra en los apéndices en la imagen E.3.

Control del motor de la cámara del robot La cámara se mueve gracias a un servomotor paso a paso cuyo rango de movimiento va de -90 a 90 grados. Este motor no está modificado, por lo que se mueve de forma habitual, según el PW que recibe se orienta hacia una dirección u otra.

Una vez entendida la relación práctica entre el ángulo de orientación y el ancho de pulso (PW), veamos cual es su relación matemática.

Como vemos en la figura 5.5, un PW de 1.5 ms equivale a la posición del motor que señala el ángulo 0, de la misma manera para que el motor esté orientado hacia el ángulo -90 deberemos mandar un PW de 0.5 ms y para orientarlo en el ángulo de 90 mandaríamos un PW de 2.5. Por lo tanto, un ms de diferencia implica 90 grados de diferencia.

Con estos datos es sencillo encontrar una fórmula de conversión de grados a PW:

$$PW = \frac{g}{90} + 1'5$$

Siendo PW el ancho de pulso que debo mandar al servomotor para que rote hasta la posición señalada por el grado g.

Asi podemos convertir los ángulos en PW y estos PW se convierten en el Duty Cycle tal y como se explica en 5.3.3 y finalmente se podrán mover los servomotores hacia la dirección deseada.

El código para el control de las ruedas del robot se encuentra en los apéndices en la imagen E.4.

6 Pruebas y Resultados

A continuación se explican las pruebas y los resultados obtenidos tras desarrollar el proyecto.

Puesto que el objetivo es crear un dispositivo sencillo de controlar por el usuario, las pruebas constarán de varias fases que se repetirán dentro de un ciclo cerrado. De esta forma el dispositivo podrá ir mejorando conforme a la demanda de los usuarios.

En la siguiente imagen se representan las diferentes fases del ciclo de pruebas.

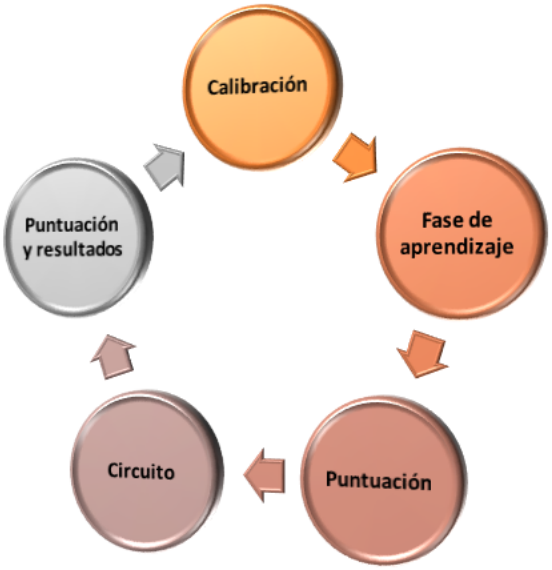


Figura 6.1: Fases del ciclo cerrado de pruebas que se han realizado para valorar la funcionalidad del dispositivo.

6.1 Fases del ciclo cerrado

A continuación se explica en qué consisten las diferentes fases del ciclo cerrado. Durante estas pruebas se realizarán tres ciclos pero solo el tercer ciclo de pruebas será completo.Las razones de esto último se explicarán a lo largo del capítulo.

- **Calibración:** Se ajustan en el código los parámetros de velocidad de los motores y sensibilidad de movimientos en base a pruebas anteriores.
  El ajuste de velocidad de los servomotores se explica en la sección 5.3.3 de este documento.
  El ajuste de la sensibilidad se hace de la siguiente forma:

  La sensibilidad se mide comparando la orientación actual de la cabeza con la orientación anterior obtenida. Si la diferencia entre ambas es muy pequeña no se envía ningún cambio de orientación al robot. Cuanto más alto fije el umbral a partir del cual envío la información, menor sensibilidad de movimiento tendrá el sistema.

  En el primer ciclo estas calibraciones se hicieron en base a mi propia experiencia utilizando el robot.

- **Fase de aprendizaje:** Se explica al usuario la interfaz y se deja que experimente con ella durante un par de minutos.

- **Puntuación:** El usuario puntúa la interfaz y la facilidad de uso del dispositivo.

- **Circuito:** Se propone al usuario un circuito que debe seguir con el robot.

- **Puntuación y resultados:** Tras realizar el circuito e interaccionar con el dispositivo durante 5/10 minutos el usuario vuelve a puntuar y propone mejoras para la nueva fase de calibración.

6.2 Desarrollo de las pruebas

El primer ciclo se realizó con un único usuario ya que la interfaz aún no estaba claramente definida.

Tras este primer ciclo se realizaron dos ciclos más hasta que se consiguieron resultados satisfactorios.

A continuación se explicarán los resultados y mejoras realizadas en cada ciclo.

6.2.1 Primer ciclo

Como ya se ha explicado anteriormente, el primer ciclo de pruebas lo realizó un solo usuario ya que en base a sus valoraciones se modificó la interfaz prácticamente por completo. El ciclo no se completó, solo se realizaron las fases de aprendizaje, puntuación y calibración. Esto se realizó así debido a que el diseño de los movimiento de cabeza para controlar el robot eran muy poco intuitivos. **En el primer ciclo de pruebas** el robot se controlaba de la siguiente forma:

- **Giros:** El robot giraba sobre sí mismo tantos grados como girara el usuario la cabeza.

- **Avanzar:** El robot avanzaba si el usuario mantenía la cabeza en posición natural, es decir, mirando hacia el frente.

- **Parar:** el robot se detenía cuando el usuario giraba la cabeza, es decir, en fase de exploración.

- **Movimiento de la cámara:** la cámara se movía tanto grados hacia arriba o hacia abajo como grados se moviera el usuario.

Valoración del usuario La primera valoración no fue muy positiva. A continuación se muestran las dificultades con las que se encontró el usuario:

- Giros: El usuario puso de manifiesto la dificultad que supone dar la vuelta al robot con la interfaz diseñada para este ciclo, ya que el usuario tendría que girar sobre sí mismo.

  Solución: Modificar el giro de forma que el robot gire con velocidad continua hacia la derecha o izquierda según la orientación de la cabeza del usuario.

- Condición de parada: Otra dificultad con la que se encontró el usuario fue que la forma de parar el robot (girando la cabeza) era muy poco intuitiva. A esto se le añade la dificultad de que el robot esté avanzando por defecto.

  Solución: Modificar el estado por defecto del dispositivo, en la nueva interfaz el robot está parado por defecto y en caso que esté en movimiento basta con mirar al frente para detenerlo.

En base a estas dificultades se ideó la nueva interfaz de control, que se explica anteriormente en este documento, en la sección 5.3.1.

6.2.2 Segundo ciclo

Durante el segundo ciclo de pruebas participaron 7 usuarios. La fase de aprendizaje durante este ciclo consistió en: girar a la derecha, girar a la izquierda, mirar hacia arriba, estar 30 segundos aproximadamente girando el robot sobre sí mismo hasta tener controlado el giro, andar hacia delante, parar, estar cerca de un minuto paseando para familiarizarse con los movimientos, fijar un objetivo y alcanzarlo.

La tabla D.1 de los apéndices muestra la valoración de los usuarios tras realizar las pruebas.

Comentarios tras las pruebas Analizando las puntuaciones y los comentarios de los usuarios se sacaron las siguientes conclusiones:
Se debía reducir la velocidad de giro ya que era complicado controlarlo debido a la velocidad. Por otro lado la nueva interfaz de control del robot que se ideó tras el primer ciclo de pruebas resultó mucho más intuitiva para los usuarios y su sensación de control incrementaba en muy pocos minutos.

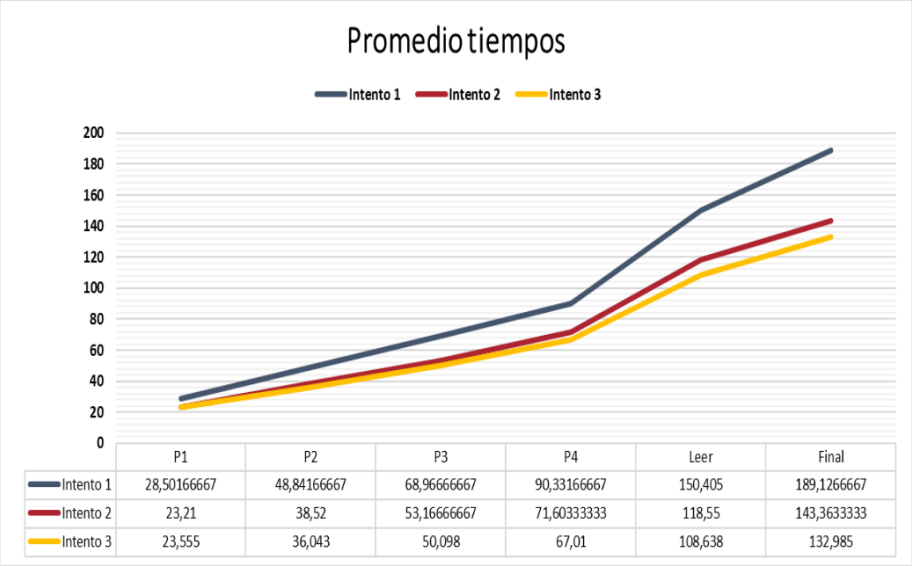
6.2.3 Tercer ciclo

A continuación se explicará el desarrollo del tercer y último ciclo de pruebas:

- **Fase de calibración:** Durante esta fase se realizaron los siguientes cambios en base a la opinión de los usuarios que participaron en el segundo ciclo de pruebas:
  - Se disminuyó la velocidad de giro de los servomotores de las ruedas del robot.
  - Se disminuyó la velocidad de avance del robot.
  - Se disminuyó la sensibilidad de detección de movimiento vertical, reduciendo las posibles posiciones que puede tomar la cámara.
- **Fase de aprendizaje:** Se le explicó a cada usuario la forma de controlar el robot y se les dejó durante dos minutos moverse libremente.
- **Puntuación:** En la tabla D.2 de los apéndices se muestra la primera puntuación de los usuarios.

- **Circuito:** Los usuarios realizan el mismo circuito 3 veces, en la gráfica 6.2 se denominan *intentos*. En el circuito se marcan 4 puntos críticos y se cronometra el tiempo que tarda cada usuario en alcanzar dichos puntos. También se cronometra el tiempo total.

  A continuación se muestra un gráfico donde se muestran los promedios de tiempos entre todos los usuarios de cada intento de hacer el circuito.



	P1	P2	P3	P4	Leer	Final
Intento 1	28,50166667	48,84166667	68,96666667	90,33166667	150,405	189,1266667
Intento 2	23,21	38,52	53,16666667	71,60333333	118,55	143,3633333
Intento 3	23,555	36,043	50,098	67,01	108,638	132,985

Figura 6.2: Promedio del tiempo que tardan los usuarios en realizar el circuito.

- **Puntuación:** Por último se les vuelve a pedir a los usuarios que valoren el control del robot. Los resultados de esta puntuación se encuantran en los apéndices, en la tabla D.3

  Respecto a la valoración de los usuarios es muy interesante ver cómo varía la sensación de control que tienen los usuarios cuando empiezan a utilizar el dispositivo y cuando ya han realizado 3 veces el circuito. Dicha variaciónn se ve reflejada en el siguiente gráfico.

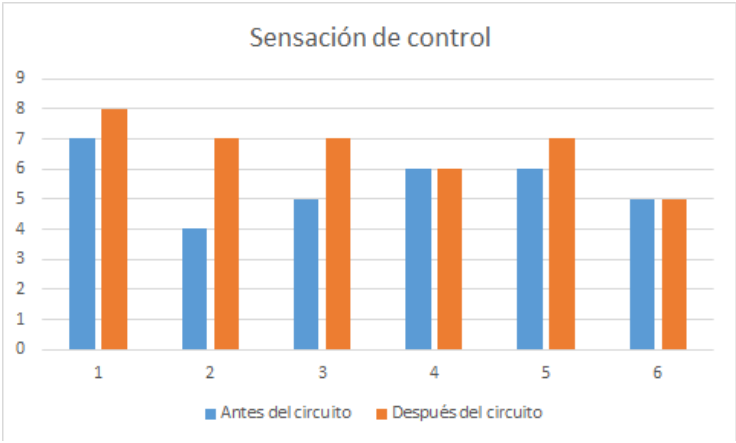


Figura 6.3: Variación de la sensación de control de los usuarios antes y después de realizar el circuito.

7 Conclusiones y trabajo futuro

7.1 Conclusiones

A modo de conclusión se presentarán las metas alcanzadas en este trabajo con las cuales se logran los objetivos propuestos. Posteriormente se analizarán los resultados obtenidos y en base a dichos resultados se estudiarán posibles mejoras para un trabajo futuro.

- Se ha diseñado y construido un **dispositivo móvil con cámara integrada** con capacidad para grabar vídeo y transmitirlo con una latencia aceptable. A su vez este dispositivo tiene la capacidad de desplazarse en todas las direcciones del plano y puede orientar la cámara rotándola respecto al eje horizontal.
- Se ha desarrollado un sistema de **transmisión y recepción de comandos** entre el usuario y el robot o dispositivo móvil.
  Este sistema se basa en comunicaciones cliente-servidor entre el ordenador, que actúa como servidor y la Raspberry Pi 3, que actúa como cliente.
- Se ha desarrollado un sistema de **transmisión y recepción de vídeo a tiempo real** entre las gafas de realidad virtual Oculs Rift DK2 y la cámara Logitech conectada a la Raspberry Pi 3.
  Este sistema se basa en comunicaciones cliente-servidor entre la Raspberry Pi 3, que actúa como servidor y el ordenador, que actúa como cliente y que una vez que recoge el streaming lo reproduce en las gafas de realidad virtual.
- Se ha desarrollado un **software de recepción y procesamiento de información** de la orientación de las Oculus Rift.
- Se ha ideado un **sistema intuitivo de movimientos de cabeza** que sirven como **comandos de control** del movimiento del robot.
- Todo el sistema ha sido probado por usuarios que han valorado de forma positiva la experiencia y todos ellos han conseguido superar las pruebas propuestas.
  Se han realizado 3 ciclos de pruebas ya que, en base a la puntuación de los usarios, se ha cambiado la interfaz de control y ajustado los parámetros pertinentes para hacer el sistema más fácil e intuitivo.

Tras repasar a grandes rasgos el trabajo realizado se exponen las posibles mejoras del proyecto en base a los resultados obtenidos en la fase de pruebas.

Puntos positivos y a mejorar Tras las pruebas es claro que la mayor dificultad con la que se encuentran los usuarios es la **velocidad de giro** de los motores de las ruedas.

La calibración de servomotores es una tarea que se resuelve de forma experimental. La dificultad que nos encontramos es que la velocidad de giro cambia según en la superficie en la que se encuentre el robot por lo que habría que realizar las pruebas en diferentes entornos y hacer una calibración media.
Con unos motores de mayor calidad los movimientos podrían ser más precisos pero la precisión obtenida ha resultado bastante aceptable para los usuarios.
Otro problema es el **alcance de red de la Raspberry Pi 3**. Si el robot se alejaba mucho del router, el streaming tenía algo más de latencia. Esto tiene fácil solución, incluyendo una tarjeta de red en la Raspberry Pi 3, con el alcance deseado.

A pesar de todos estos puntos a mejorar el resultado es muy bueno, teniendo en cuenta que es una primera aproximación a lo que que pretende ser un proyecto futuro.
Los usuarios han estado cada uno menos de 7 minutos con el robot y **todos menos uno lograron alcanzar su objetivo** en el tiempo estipulado, por lo que es una **aplicación fácil de manejar y de aprender**, como se ha visto en las valoraciones.

En el caso de personas con discapacidad motora no se han podido hacer las pruebas ya que esta primera toma de contacto resultaba más sencilla con personas sin discapacidad, pero se espera que parámetros como la velocidad o el rango de movimiento de la cabeza sean diferentes, por lo que el sistema está diseñado para poder ajustar estos parámetros según las necesidades del usuario. La idea es realizar estas pruebas en un posible trabajo fin de master, como continuación de este proyecto.

Ha sido un proyecto con una **muy buena acogida**, a todos los usuarios les ha resultado ameno hacer las pruebas y ver cómo iban mejorando.
También es fácil de desarrollar con las herramientas de las que disponemos y tiene **mucho potencial**, ya que incluye dos tecnologías que están en pleno desarrollo y que aún tienen mucho que avanzar.
Otra ventaja es que hace uso de recursos y herramientas de **fácil acceso** por lo que no resultaría demasiado caro de fabricar.

7.1.1 Tecnologías aprendidas

Durante la realización de este proyecto he utilizado muchas tecnologías, alguna de ellas completamente nuevas para mi.

El ejemplo más claro de esto son las gafas de realidad virtual Oculus Rift. Ha sido una suerte tener acceso a ellas y poder aprender a utilizar algunas de las librerías que hay disponibles para los desarrolladores.

También ha sido una sorpresa descubrir la gran variedad de posibilidades a desarrollar que permiten las Oculus, como es este proyecto.
He afianzado los conocimientos de robótica que ya tenía. Todo lo que sabía de robótica lo había aprendido en el Club de Robótica de esta universidad y gracias a este trabajo he podido aplicarlos y aumentarlos.

Todo el proyecto está desarrollado en python, lenguaje de programación que solo había visto muy por encima para proyectos sencillos.
Durante este proyecto he aprendido a utilizar la API de python disponible para las Oculus Rift y las librerías para el control de servomotores.

Tampoco había tenido la oportunidad de utilizar nunca la placa base del robot, la Raspberry Pi 3, de todas formas esto no ha sido completamente novedoso para mi porque sí había participado en algún proyecto sencillo con Raspberry.

En conclusión este trabajo ha sido muy instructivo y ha ampliado de forma significativa mis conocimientos sobre robótica y realidad virtual.

7.2 Trabajo futuro

Este trabajo es solo el inicio de lo que podría ser un proyecto muy interesante.

El robot que hemos construido es una extensión del sentido de la vista, pero se podría construir un robot controlado por una persona y que fuera la extensión de todos sus sentidos, permitiendo así conocer y sentir el mundo sin necesidad de moverse de su casa.

Además la realidad virtual es una herramienta muy potente, que no solo te permite modificar tu mundo a placer sino que te permita crear entornos totalmente distintos. Se podría estudiar cómo interactuar con ese mundo virtual no solo a través de la vista, sino con todos los sentidos.
Un objetivo básico para la continuación de éste trabajo es la capacidad de autocalibración, es decir, que el sistema pueda modificar los parámetros de movimiento y control del robot de forma automática, basándose en pruebas y medidas anteriores. Esto es necesario ya que cada persona con discapacidad motora tendrá unas capacidades motrices distintas y posiblemente dentro de unos rangos muy amplios.
Otro objetivo con vistas a futuro es desarrollar una interfaz gráfica para toda la aplicación ya que por ahora todo arranca a través de líneas de comandos. Además es sencillo dotar a esta posible interfaz gráfica de la capacidad de cambiar los parámetros de calibración, por lo que cada usuario podrá crearse un "perfil" en el sistema.

Otro camino por el que se puede investigar es la unión de este proyecto con la domótica. Creo que en un futuro no muy lejano, las casas serán casas domotizadas, que igual que se podrán controlar desde el móvil podremos controlarlas con las gafas de realidad virtual.

BIBLIOGRAFÍA

[1] Changyin Zhou. How bad are Virtual Reality headsets for your eyes. https://www.quora.com/How-bad-are-Virtual-Reality-headsets-for-your-eyes#, 2015.

[2] Ivan E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, AFIPS '68 (Fall, part I), pages 757–764, New York, NY, USA, 1968. ACM.

[3] Hunter G. Hoffman, David R. Patterson, Jeff Magula, Gretchen J. Carrougher, Karen Zeltzer, Stephen Dagadakis, and Sam R. Sharar. Water-friendly virtual reality pain control during wound care. *Journal of Clinical Psychology*, 60(2):189–195, 2004.

[4] F Tahriri, M Mousavi, HJ Yap, MD Siti Zawiah, and Z Taha. Optimizing the robot arm movement time using virtual reality robotic teaching system. *International Journal of Simulation Modelling*, 14(1):28–38, 2015.

[5] Xataka. Los mejores juegos de realidad virtual para 2016. https://www.xataka.com/videojuegos/los-mejores-juegos-de-realidad-virtual-para-2016-se-estan-cocinando-ahora, 2015.

[6] Robert Leeb, Doron Friedman, Mel Slater, and Gert Pfurtscheller. A tetraplegic patient controls a wheelchair in virtual reality. In *BRAINPLAY 07 Brain-Computer Interfaces and Games Workshop at ACE (Advances in Computer Entertainment) 2007*, page 37, 2012.

[7] David R Hedgley. *A general solution to the hidden-line problem*, volume 1085. Citeseer, 1982.

[8] D.F. Kocian, AEROSPACE MEDICAL RESEARCH LAB WRIGHT-PATTERSON AFB OHIO., and Air Force Aerospace Medical Research Laboratory (U.S.). *A Visually-Coupled Airborne Systems Simulator (VCASS): An Approach to Visual Simulation*. Aerospace Medical Research Laboratories, 1977.

[9] Delphine Lamargue-Hamel, Mathilde Deloire, Aurore Saubusse, Aurélie Ruet, Jacques Taillard, Pierre Philip, and Bruno Brochet. Cognitive evaluation by tasks in a virtual reality environment in multiple sclerosis. *Journal of the Neurological Sciences*, 359(1–2):94 – 99, 2015.

[10] Fei Hu. *Virtual Reality Enhanced Robotic Systems for Disability Rehabilitation*. IGI Global, 2016.

[11] Rahul R Kaliki, Rahman Davoodi, and Gerald E Loeb. Evaluation of a noninvasive command scheme for upper-limb prostheses in a virtual reality reach and grasp task. *IEEE Transactions on Biomedical Engineering*, 60(3):792–802, 2013.

[12] Roberto Lloréns, Enrique Noé, Carolina Colomer, and Mariano Alcañiz. Effectiveness, usability, and cost-benefit of a virtual reality–based telerehabilitation program for balance recovery after stroke: A randomized controlled trial. *Archives of Physical Medicine and Rehabilitation*, 96(3):418 – 425.e2, 2015.

[13] James Patton, Greg Dawe, Chris Scharver, Ferdinando Mussa-Ivaldi, and Robert Kenyon. Robotics and virtual reality: a perfect marriage for motor control research and rehabilitation. *Assistive Technology*, 18(2):181–195, 2006.

[14] Laurent A. Nguyen, Maria Bualat, Laurence J. Edwards, Lorenzo Flueckiger, Charles Neveu, Kurt Schwehr, Michael D. Wagner, and Eric Zbinden. Virtual reality interfaces for visualization and control of remote vehicles. *Autonomous Robots*, 11(1):59–68, 2001.

[15] Michael Connolly, Johnathan Seligman, Andrew Kastenmeier, Matthew Goldblatt, and Jon C Gould. Validation of a virtual reality-based robotic surgical skills curriculum. *Surgical endoscopy*, 28(5):1691–1694, 2014.

[16] G. Hubens, H. Coveliers, L. Balliu, M. Ruppert, and W. Vaneerdeweg. A performance study comparing manual and robotically assisted laparoscopic surgery using the da vinci system. *Surgical Endoscopy And Other Interventional Techniques*, 17(10):1595–1599, 2003.

[17] Dexterous Observational Roving Automaton. http://doraplatform.com/.

[18] Jesus Savage-Carmona, Mark Billinghurst, and Alistair Holden. The virbot: a virtual reality robot driven with multimodal commands. *Expert Systems with Applications*, 15(3):413–419, 1998.

[19] Alexandre Monferrer and David Bonyuet. Cooperative robot teleoperation through virtual reality interfaces. In *Information Visualisation, 2002. Proceedings. Sixth International Conference on*, pages 243–248. IEEE, 2002.

[20] Carlos García-Saura and Juan González-Gómez. Low cost educational platform for robotics, using open-source 3d printers and open-source hardware. In *ICERI2012 Proceedings*, pages 2699–2706. IATED, 2012.

CONFIGURACIÓN DE RED LOCAL

A

En el apéndice A se encuentran las imágenes que muestran la configuración del router y de la Raspberry Pi para conectarse automáticamente a la red local.

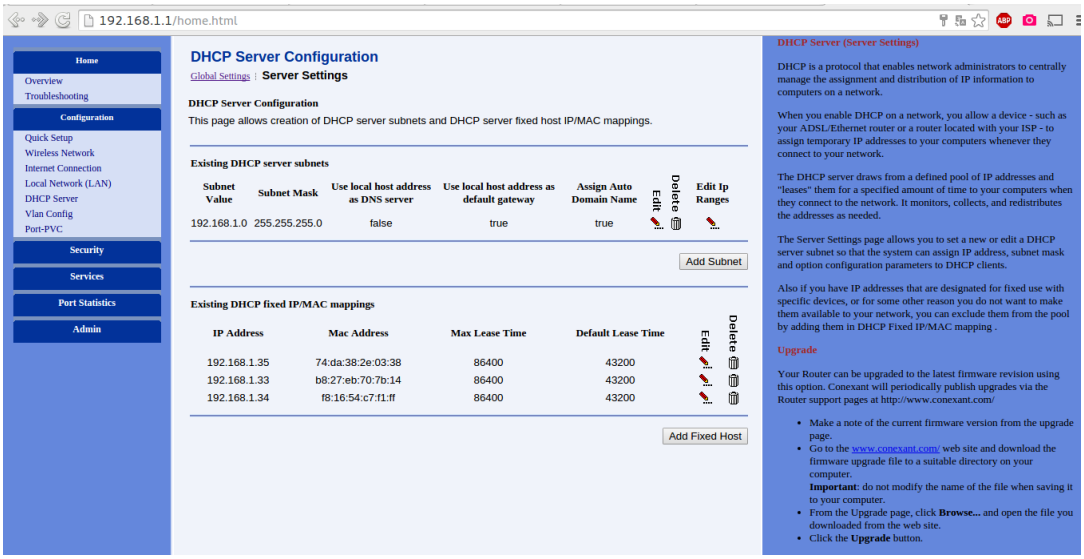


Figura A.1: Tabla DHCP del router con las parejas MAC-IP fijas

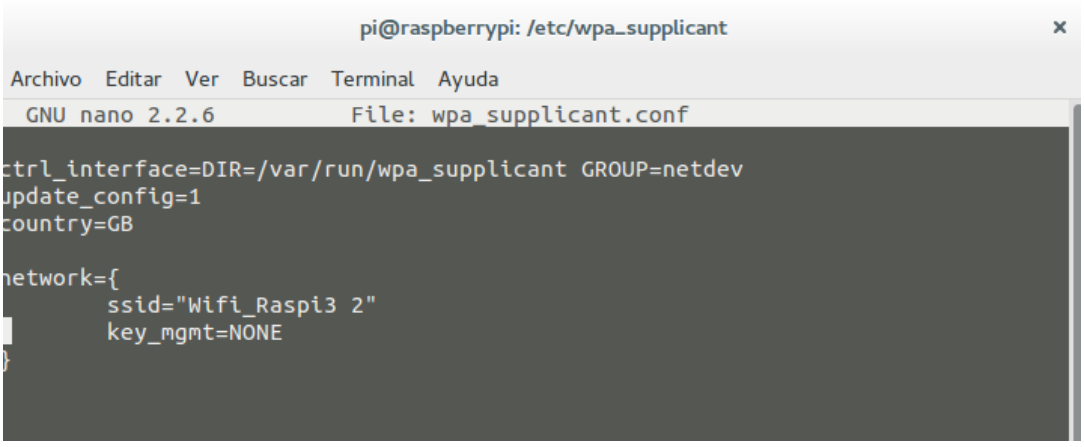
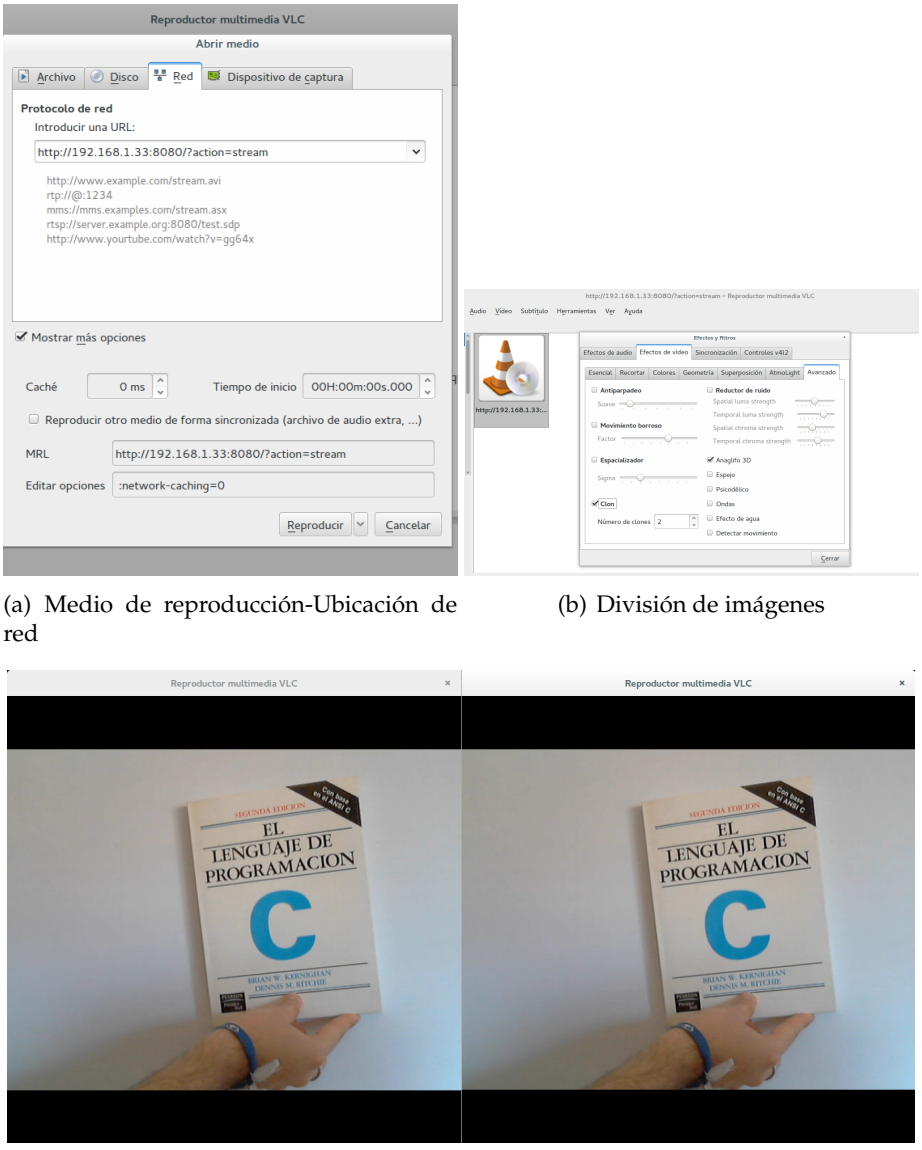


Figura A.2: Archivo de configuración de red Raspberry Pi 3

CONFIGURACIÓN DE VLC

B

En la siguiente imagen se ve la forma de configurar el reproductor de vídeo VLC para poder dividir la imagen del streaming en dos.



(a) Medio de reproducción-Ubicación de red

(b) División de imágenes

(c) Reproducción SBS

Figura B.1: Configuración y resultado de la reproducción de vídeo en VLC

Componentes

C

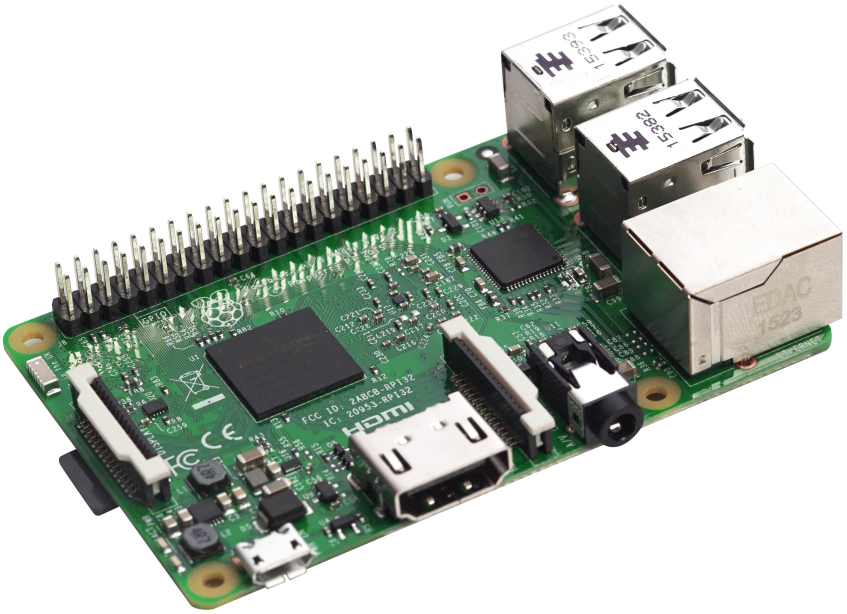


Figura C.1: Raspberry Pi 3 model B

PRUEBAS

D

En el apéndice B se adjuntan las imágenes referentes a las pruebas realizadas para este trabajo.

La imagen D.1 muestra el circuito que tenian que realizar los usuarios durante el tercer ciclo de pruebas.
Se puede ver cómo el circuito tiene 4 curvas, estas curvas son los puntos P1, P2, P3, P4 que se miden en la fase de pruebas 6.2.

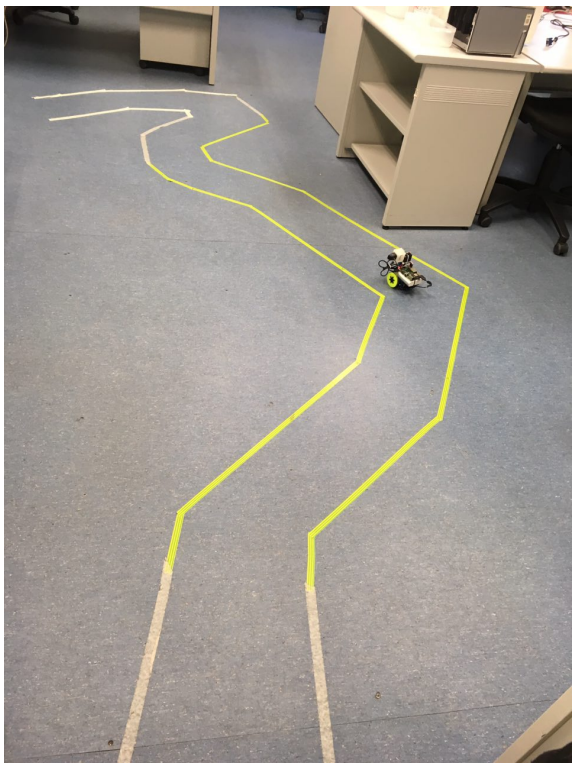


Figura D.1: Circuito que deben seguir los usuarios en las pruebas.

A continuación se añaden las tablas con la valoración que hicieron los usuarios que realizaron el segundo ciclo de pruebas y los usuarios qeu realizaron el tercer ciclo de pruebas tras completrar las mismas.

Valoración de los usuarios en el segundo ciclo							
	Usuario1	Usuario2	Usuario3	Usuario4	Usuario5	Usuario6	Usuario7
Giro derecha	9	6	6	4	5	5	6
Giro izquierda	7.5	4	6	4	5	5	7
Mirar hacia arriba	9	10	9	10	9	9	9
Andar	10	8	10	8	8	9	9
Parar	9	10	9	7	7	7	8
Alcanzar un objetivo	8.5	6	8	7	5	5	8
Sensación de Control	7.5	6	9	4	5	4	7
Facilidad de aprendizaje	9	8	10	10	9	8	10

Tabla D.1: Resultado de los test de usabilidad realizados por los usuarios al finalizar el segundo ciclo.

Valoración de los usuarios tras la fase de aprendizaje del tercer ciclo						
	Usuario1	Usuario2	Usuario3	Usuario4	Usuario5	Usuario6
Avanzar	8	7	8	9	6	8
Giro derecha	6	5	6	7	7	6
Giro izquierda	5	5	5	5	7	5
Telepresencia	9	8	8	7	5	6
Sensación de control	7	4	5	6	6	5
Interfaz	9	7	7	9	6	9

Tabla D.2: Resultado de los test de usabilidad realizados por los usuarios que participaron en el tercer ciclo de pruebas una vez finalizada la fase de aprendizaje

Valoración de los usuarios tras realizar el circuito 3 veces						
	Usuario1	Usuario2	Usuario3	Usuario4	Usuario5	Usuario6
Avanzar	8	9	8	6	6	8
Giro derecha	7	7	7	8	6	6
Giro izquierda	7	7	6	8	6	5
Telepresencia	9	8	9	7	6	6
Sensación de control	8	7	7	6	7	5
Interfaz	9	8	7	9	6	9

Tabla D.3: Resultado de los test de usabilidad realizados por los usuarios que participaron en el tercer ciclo de pruebas tras realizar el circuito

CÓDIGO DE SOFTWARE DE CONTROL DEL ROBOT

E

```
    time.sleep(5)
    ts  = ovr.getTrackingState(session, ovr.getTimeInSeconds(), True)
    if ts.StatusFlags & (ovr.Status_OrientationTracked | ovr.Status_PositionTracked):

        time.sleep(0.200)
        kat_x_aux, kat_y_aux = get_angles(ts.HeadPose.ThePose.Orientation);


    while(1):
        ts  = ovr.getTrackingState(session, ovr.getTimeInSeconds(), True)
        if ts.StatusFlags & (ovr.Status_OrientationTracked | ovr.Status_PositionTracked):

        kat_x, kat_y = get_angles(ts.HeadPose.ThePose.Orientation);

            if(abs(kat_x_aux-kat_x)>K_SENS_X):
                if((kat_x >= -90.0) and (kat_x <=90.0)):
                    if (kat_x < -30.0):
                        q2.send("A")
                        kat_x=-20.0
                        data=str(kat_x)
                        q.send(data)
                    elif (kat_x > -25.0):
                        q2.send("P")
                        data=str(kat_x)
                        q.send(data)
                    else:
                        data = str(kat_x)
                        q.send(data)
                    kat_x_aux = kat_x
            kat_x = 0


            if(abs(kat_y_aux-kat_y)>K_SENS_Y):
                data = str(kat_y)
                q2.send(data)
                kat_y_aux = kat_y
            kat_y = 0

        time.sleep(K_SLEEP)

    ovr.destroy(session)
    ovr.shutdown()
except:
        print "Init error"
        ovr.destroy(session)
        ovr.shutdown()
```

Figura E.1: Código que se ejecuta en el lado del PC para leer la información de las Oculus Rift, procesarla y mandarla a la Raspberry Pi 3.

En el siguiente código E.1 se ve cómo el PC recoge la información de la orientación de las Oculus Rift y llama a la función `get_angles`, que le devuelve el valor del ángulo respecto al eje horizontal y el ángulo respecto al eje vertical. Según dichos ángulos el PC le manada la información pertinente a la Raspberry.

A continuación se incluye el código de la función `get _angles`.

```
def get_angles(quat_oc):

    a = quat_oc.x;
    b = quat_oc.y;
    c = quat_oc.z;
    d = quat_oc.w;

    x_angle = np.arctan2(2 * (a*d + b*c), 1 - 2 * (c*c + d*d));
    y_angle = np.arcsin(2 * (a*c - d*b));


    if x_angle == 180:
        x_angle = 0;

    elif x_angle < 0:
        x_angle = (-1)*(180+x_angle);

    else:
        x_angle = 180 - x_angle;

    return x_angle , y_angle;
```

Figura E.2: Función que se ejecuta en el PC para transformar la información de las Oculus Rift en ángulos respeco a los ejes horizontal y vertical.

Una vez procesada la información en el PC se envía al robot. A continuación se incluyen los códigos que se ejecutan en la Raspberri Pi 3, tanto el códigop que controla las ruedas del robot E.3 como el que controla el movimiento de la cámara E.4.

```
host = "192.168.1.36"

print host

port=4447

s=socket(AF_INET, SOCK_STREAM)

print "socket made"

s.connect((host,port))

print "socket connected!!!"


GPIO.setmode(GPIO.BOARD)

GPIO.setup(16,GPIO.OUT)
pD = GPIO.PWM(16,50)
pD.start(50)

GPIO.setup(18,GPIO.OUT)
pI = GPIO.PWM(18,50)
pI.start(50)


pD.ChangeDutyCycle(0)
pI.ChangeDutyCycle(0)
dif = 90.0
#msg=s.recv(1024)
pD.ChangeDutyCycle(0)
pI.ChangeDutyCycle(0)

try:
    while True:      #iniciamos un loop infinito
        try:

                msg=s.recv(1024)

                if(msg=="A"):
                        pD.ChangeDutyCycle(55.9)
                        pI.ChangeDutyCycle(5.85)
                elif(msg=="P"):
                         pD.ChangeDutyCycle(0)
                         pI.ChangeDutyCycle(0)
                else:

                        currH=float(msg)
                        if currH>=-15.0 and currH<=15.0:
                                pD.ChangeDutyCycle(0)
                                pI.ChangeDutyCycle(0)

                        elif currH< -15.0:
                                # izquierda
                                pI.ChangeDutyCycle(7.1)
                                pD.ChangeDutyCycle(7.55)
                        elif currH> 15.0:
                                #derecha
                                pI.ChangeDutyCycle(6.4)
                                pD.ChangeDutyCycle(6.7)

        except ValueError:
                print("ValueError")
except KeyboardInterrupt:         #CONTROL+C ...
        pI.stop()
        pD.stop()
        GPIO.cleanup()
```

Figura E.3: Código que se ejecuta en la Raspberry Pi 3 para controlar las ruedas del robot.

```
from socket import *
import RPi.GPIO as GPIO
import time
import datetime
import numpy as np

def angle_to_PWM(angle):
        print (angle)
        if(angle>=70 and angle <=90):
                pulse = 0.62
        elif(angle>=50 and angle<70):
                pulse = 0,83
        elif(angle>= 30 and angle < 50):
                pulse = 1
        elif(angle>=10 and angle<30):
                pulse = 1.3
        elif(angle>=-10 and angle < 10):
                pulse =1.50
        elif(angle>=-30 and angle < -10):
                pulse =1.70
        else:
                pulse = 1.8
        #dutycycle
        dcycle = (pulse/20)*100
        return dcycle

host = "192.168.1.36"

print host

port=4446

s=socket(AF_INET, SOCK_STREAM)

print "socket made"

s.connect((host,port))

print "socket connected!!!"

GPIO.setmode(GPIO.BOARD)
GPIO.setup(12,GPIO.OUT)
p = GPIO.PWM(12,50)
p.start(5)



try:
    while True:
        try:

                msg=s.recv(1024)
                dcycle =float( angle_to_PWM(float(msg)))
                p.ChangeDutyCycle(dcycle)

        except ValueError:
                print("ValueError")
except KeyboardInterrupt:         #CONTROL+C ...
        p.stop()
```

Figura E.4: Código que se ejecuta en la Raspberry Pi 3 para controlar el servomotor unido a la cámara.